ORF 555 HW 5

(Q1) 1-period Bitcoin mining:-

expend → $\alpha \geq 0$; prob of winning $(p) = \dfrac{\alpha}{\alpha + A}$

$$\underbrace{\text{Gain/loss}}_{\text{profit } (P_r)} = rZ - c\alpha = \begin{cases} r - c\alpha & \text{wp } p \\ -c\alpha & \text{wp } 1-p \end{cases}$$

$$E(P_r) = p(r - c\alpha) + (1-p)(-c\alpha) = rp - c\alpha$$

$$= \frac{r\alpha}{\alpha + A} - c\alpha = f(\alpha)$$

(a) $\alpha^* \Rightarrow \dfrac{\partial f}{\partial \alpha} = 0 \Big|_{\alpha = \alpha^*} \Rightarrow \dfrac{r(\alpha + A - \alpha)}{(\alpha + A)^2} - c = 0 \Big|_{\alpha = \alpha^*}$

$\Rightarrow \dfrac{rA}{(\alpha^* + A)^2} = c \Rightarrow \boxed{\alpha^* = \sqrt{\dfrac{rA}{c}} - A} \rightarrow \boxed{\left(\text{if } c \leq \dfrac{r}{A};\ \text{if } c \geq \dfrac{r}{A}, \alpha^* = 0\right)}$

$\dfrac{\partial^2 f}{\partial \alpha^2}\Big|_{\alpha = \alpha^*} = \dfrac{-2rA}{(\alpha^* + A)^3} < 0 \Rightarrow \alpha^* =$ maximiser of f $(E(P_r))$

(b) $\alpha^* = 0 \Rightarrow \boxed{c^* = \dfrac{r}{A}}$ $\left(\sqrt{\dfrac{rA}{c}} - A \leq 0 \Rightarrow \sqrt{\dfrac{rA}{c}} \leq A \right.$

$\left. \Rightarrow \dfrac{rA}{c} \leq A^2 \Rightarrow c \geq \dfrac{r}{A} \right)$

(c) $M+1$ miners with different costs of electricity ...
Interacting through A
$A \sim M\bar{\alpha}$; Assuming distᵑ of $\underbrace{\text{electricity}}_{c}$ costs is by $m(c)$

$\to A \to c \to \alpha^*(c)$
$\to A = M\bar{\alpha}$
↳ Mean hash rate $\left(\bar{\alpha} = \int \alpha^*(c)\, m(c)\, dc\right)$

$\because c \in U[\frac{r}{2}, r]$,
$m(c) = \frac{2}{r} \mathbb{1}[\frac{r}{2} \le c \le r]$

(i) Note that $\alpha^*(c) = 0$ for $c > c^*$

$$\Rightarrow \bar{\alpha} = \int_{r/2}^{c^*} \alpha^*(c)\, m(c)\, dc$$

$$= \int_{r/2}^{r/A} \left(\sqrt{\frac{rA}{c}} - A\right) \frac{2}{r}\, dc$$

$$= \frac{2}{r}\sqrt{rA} \int_{r/2}^{r/A} \frac{1}{\sqrt{c}}\, dc - \frac{2A}{r} \int_{r/2}^{r/A} dc$$

$$= 2\sqrt{\frac{A}{r}} \times 2\left(\sqrt{\frac{r}{A}} - \sqrt{\frac{r}{2}}\right) - \frac{2A}{r}\left(\frac{r}{A} - \frac{r}{2}\right)$$

$$\cancel{= 4 - 4\sqrt{\frac{A}{2}} - 2 + A}$$

$$= 4\sqrt{\frac{A}{r}}\left(\sqrt{\frac{r}{A}} - \sqrt{\frac{r}{2}}\right) - \frac{2A}{r}\left(\frac{r}{A} - \frac{r}{2}\right)$$

$$= 4 - 4\sqrt{\frac{A}{2}} - 2 + A$$

$$= 2 - 2\sqrt{2}\sqrt{A} + A$$

$$= (\sqrt{2} - \sqrt{A})^2$$

$$\Rightarrow \boxed{\bar{\alpha} = (\sqrt{2} - \sqrt{A})^2}$$

So, $A = M\bar{\alpha}$

$= M(\sqrt{2}-\sqrt{A})^2$

$\Rightarrow \sqrt{A} = \sqrt{M}(\sqrt{2}-\sqrt{A})$

$\cancel{\Rightarrow \sqrt{M} = \frac{\sqrt{A}}{\sqrt{2}-\sqrt{A}}}$

$\Rightarrow (1+\sqrt{M})\sqrt{A} = \sqrt{2M}$

$\Rightarrow \sqrt{A} = \frac{\sqrt{2M}}{1+\sqrt{M}}$

$$\Rightarrow \boxed{A = \frac{2M}{(1+\sqrt{M})^2}}$$

(ii) As $M \to \infty$, $A \to 2$

upper bound: But $A = \frac{2(\sqrt{M})^2}{(1+\sqrt{M})^2} = 2\left(\frac{\sqrt{M}}{1+\sqrt{M}}\right)^2 < 2$

& M large enough $\Rightarrow M \gg 1$ $\cancel{\Rightarrow \frac{1}{1+\sqrt{M}} \ll 1}$

$\Rightarrow \frac{\sqrt{M}}{1+\sqrt{M}} \sim 1$

$\Rightarrow \underline{A \sim 2}$

$\rightarrow$ Specifically,

$A > 1 \Rightarrow 2\left(\frac{\sqrt{M}}{1+\sqrt{M}}\right)^2 > 1$

lower bound: $\Rightarrow \left(\frac{\sqrt{M}}{1+\sqrt{M}}\right)^2 > \frac{1}{2} \Rightarrow \frac{\sqrt{M}}{1+\sqrt{M}} > \frac{1}{\sqrt{2}}$

$\Rightarrow \frac{1}{1+\sqrt{M}} < 1 - \frac{1}{\sqrt{2}} = \frac{\sqrt{2}-1}{\sqrt{2}}$

~~lower~~

$\Rightarrow 1+\sqrt{M} > \frac{\sqrt{2}}{\sqrt{2}-1} = \sqrt{2}(\sqrt{2}+1) = 2+\sqrt{2}$

$\Rightarrow \sqrt{M} > 1+\sqrt{2}$

$\Rightarrow M > 1+2+2\sqrt{2} = 3+2\sqrt{2} \approx 5.8$

$\because$ M is large, $M > 6 > 5.8$ ✓

(iii) $c^* = \frac{r}{A} = \boxed{\frac{r}{2M}(1+\sqrt{M})^2}$ (As $M \to \infty$, $c^* \to \frac{r}{2}$)

(iv) fraction of miners that are active are those that have cost $< c^*$

$\Rightarrow f = \int_{r/2}^{c^*} m(c)\,dc$

Already density $\Rightarrow < 1$
$\Rightarrow$ No need to divide again by M ✓

| But is $\int_{r/2}^{c^*} m(c)\,dc = M$?

$m(c)$ ↳ distⁿ of c
↳ continuum of miners over cost c

No since $\int_{r/2}^{c^*} m(c)\,dc < \int_{r/2}^{r} m(c)\,dc = 1$]

$\int_{r/2}^{c^*} \frac{2}{r}\,dc = (c^* - r/2)\,2/r$

$= c^* \cdot \frac{2}{r} - 1$

$= \frac{r}{A} \cdot \frac{2}{r} - 1$

$= \frac{2}{A} - 1$

$= \frac{(1+\sqrt{M})^2}{M} - 1$

Essentially, f = ~~[r/2, c*]~~ on [r/2 ——— c* ——— r]

⇓

$$f = \frac{c^* - r/2}{r - r/2}$$

$$= \frac{r/A - r/2}{r/2}$$

$$= \frac{2}{A} - 1$$

$$= \frac{(1+\sqrt{M})^2}{M} - 1$$

$$= \frac{M + 2\sqrt{M} + 1}{M} - 1$$

$$= \cancel{1} + \frac{2}{\sqrt{M}} + \frac{1}{M} - \cancel{1}$$

$$= \boxed{\frac{2}{\sqrt{M}} + \frac{1}{M}}$$ (# miners $= 2\sqrt{M} + 1$)

As $M \to \infty$, $\boxed{f \to 0}$

(Q.2) Now, $p = \frac{\alpha x}{\alpha x + AX}$ (Here, α (most probably) represents cost of elec/hash · # hashes/\$ i.e. total cost/\$)

H (profit) $= rz - \alpha$

$$E(H) = r\,E(z) - \alpha = rp - \alpha = \frac{r\alpha x}{\alpha x + AX} - \alpha$$

$$= r\left(1 - \frac{AX}{\alpha x + AX}\right) - \alpha$$

$$= r - \frac{rAX}{\alpha x + AX} - \alpha$$

$$= (r - \alpha) - \frac{rAX}{\alpha x + AX} = f(\alpha) \quad - (*)$$

(a)

$\frac{\partial f}{\partial \alpha} = 0$ at $\alpha = \alpha^*$ $\Rightarrow$ $-1 + \frac{rAX \cdot x}{(\alpha^* x + AX)^2} = 0$

$$\Rightarrow \frac{rAXx}{(\alpha^* x + AX)^2} = 1$$

$$\Rightarrow \alpha^* x + AX = \sqrt{rAXx}$$

$$\Rightarrow \boxed{\alpha^* = \frac{\sqrt{rAXx} - AX}{x}}$$

(if $d = \frac{x}{X} \geq \frac{A}{r}$; if $d \leq \frac{A}{r}$, $\alpha^* = 0$)

(b) Let $\frac{x}{X} = d$

$$\Rightarrow \alpha^* = \sqrt{\frac{rA}{d}} - \frac{A}{d} \leq 0 \Rightarrow \sqrt{\frac{rA}{d}} \leq \frac{A}{d} \Rightarrow \frac{rA}{d} \leq \frac{A^2}{d^2}$$

$$\Rightarrow \boxed{d \leq \frac{A}{r}}$$

(c) As Individual wealth (x) increases, firstly $\alpha^* \uparrow$ (more chance of α^* being positive now) & also due to $x \uparrow$ & $\alpha \uparrow$, (assuming A & X are constant), p (prob. reward) $\uparrow$

So, the one with more wealth has more/higher chance of getting reward (discovering bitcoin) ~~More wealth~~ due to which the wealth gap/ inequality is likely to increase. This supports the idea of preferential attachment.

Hence, instead of getting decentralized (where wealth ~~distribution~~ inequality decreases & distribution is more uniform among players), here wealth inequality is increasing & so ~~goes~~ leads us to be against the idea of decentralized finance.

Q.3) Financialization. α^n

$$\rho_t = \epsilon \frac{\theta \pi \sigma_s x_t}{A\, \sigma_t^{(0)}}$$

(a) If A is very large, $\rho_t \to 0 \Rightarrow$ financialization correlation tends to 0.

In general, $\boxed{\rho_t \propto \frac{1}{A}}$ (Inversely proportional to A).

So, as $A \uparrow$, $\rho_t \downarrow$.

$\Rightarrow$ ~~compared to sup~~ If supply: A is small, ρ_t is large
If A is large, ρ_t is small.

$\Rightarrow$ compared to supply being less, if supply is more, corr^n is less.

So, if A is large, ρ_t = small
If A is small, ρ_t = large (significant).

So, compared to A being ~~large~~ small, if $\underset{\text{supply}}{A}$ were large,
the financialization correlation (ρ_t) would decrease (be less).

Why? If supply is smaller, every unit of the supply matters (i.e. marginal utility of the units is high); So, correlation would kick on. However, if the supply is large, the impact of speculators does not affect the actual trading or market of the commodity, thus decreasing ρ_t.

supply small

- → small market
- → impact of speculators high (how they trade)
- → s_t high

supply large

- → large market
- → spec. impact low
- → hence, s_t low.

(b) Note that $\theta + \pi \le 1$ ($\because$ unleveraged)

To maximize $s_t \propto \theta\pi$, $\boxed{\theta = \pi = \frac{1}{2}}$

(Q. 4)

(a) $\log Y \sim N(\mu, \sigma^2)$ under risk-neutral prob Q $\quad$ $\log Y \in (-\infty, \infty) \Rightarrow Y \in (e^{-\infty}, e^{\infty}) = (0, \infty)$

So, $E^Q\{(1-Y)^+\} = \int_0^\infty (1-y)^+ f_Y^Q(y)\,dy = \int_0^1 (1-y) f_Y^Q(y)\,dy + \int_1^\infty 0 \ldots$

$= \int_0^1 (1-y) f_Y^Q(y)\,dy$

$\quad Y = e^{N(\mu,\sigma^2)}$
$\quad 1-Y = 1 - e^{N(\mu,\sigma^2)}$

$= \int_0^1 f_Y^Q(y)\,dy - \int_0^1 y f_Y^Q(y)\,dy$

$= P(Y \le 1) - \left[\left(y \int f_Y^Q(y)dy\right)_0^1 - \int_0^1 \left(\int f_Y^Q(y)dy\right) dy\right]$

$= P(\log Y \le 0) - \left[\left(y F_Y^Q(y)\right)_0^1 - \int_0^1 F_Y^Q(y)\,dy\right]$

$= P(N(\mu,\sigma^2) \le 0) - \left[F_Y^Q(1)\right] + \int_0^1 F_Y^Q(y)\,dy$

$\log y = t \Rightarrow \frac{dy}{y} = dt$

$\Rightarrow I = \int_{-\infty}^0 (1-e^t) f_T^Q(t)\,dt = \int_{-\infty}^0 f_T^Q(t)\,dt - \int_{-\infty}^0 e^t f_T^Q(t)\,dt$

So, $E^Q((1-Y)^+) \cancel{=} - E^Q((Y-1)^+)$ $\quad$ $| \; \cancel{dX = \mu dt + \sigma dz}$

$Y = \mu + \sigma Z \to N(0,1)$

$(Y_T = \ln(\frac{S_T}{X_T}) = \ln(\frac{S_0}{X_0}) + (\quad)T + \sigma B_T - \eta W_T^{Q1})$

$\sigma^2 T + \eta^2 T - 2\sigma\eta S T = (\sigma^2 + \eta^2 - 2\sigma\eta S)T)$

From wikipedia, for a lognormal RV Y,

$g(K) = \int_K^\infty y f_Y^Q(y)\,dy = e^{\mu+\sigma^2/2}\phi\left(\frac{\mu - \ln K}{\sigma} + \sigma\right)$

$\Rightarrow g(1) = e^{\mu+\sigma^2/2}\phi\left(\frac{\mu+\sigma^2}{\sigma}\right)$

$\Rightarrow \int_0^1 y f_Y^Q(y)\,dy = E(Y) - g(1) = e^{\mu+\sigma^2/2} - e^{\mu+\sigma^2/2}\phi\left(\frac{\mu+\sigma^2}{\sigma}\right)$

$= e^{\mu+\sigma^2/2}\phi\left(\frac{-\mu-\sigma^2}{\sigma}\right)$

$\Rightarrow$ Final answer $= E^Q((1-Y)^+) = \boxed{\phi\left(-\frac{\mu}{\sigma}\right) - e^{\mu+\sigma^2/2}\phi\left(-\frac{\mu}{\sigma} - \sigma\right)}$

How? If $X = LN(\mu, \sigma^2)$ (log normal) $\Rightarrow$ & $\ln X = Y \sim N(\mu, \sigma^2)$

$$\int_k^\infty x\, f_X(x)\, dx = E(X\, 1_{X \geq k}) = E(e^Y 1_{e^Y \geq k}) = E(e^Y 1_{Y \geq \ln k})$$

$$= \int_{\ln k}^\infty e^y f_Y(y)\, dy = \int_{\ln k}^\infty e^y \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right) dy$$

$$= \frac{1}{\sigma\sqrt{2\pi}} \int \exp\left(-\frac{(y^2 - 2\mu y + \mu^2)}{2\sigma^2} + y\right) dy$$

$$= \frac{1}{\sigma\sqrt{2\pi}} \int \exp\left(\frac{-y^2 + 2\mu y - \mu^2 + 2\sigma^2 y}{2\sigma^2}\right) dy$$

$$= \frac{1}{\sigma\sqrt{2\pi}} \int \exp\left(-\frac{[(y - (\mu+\sigma^2))^2 - (\mu+\sigma^2)^2 + \mu^2]}{2\sigma^2}\right) dy$$

$$= \frac{1}{\sigma\sqrt{2\pi}} \int \exp\left(-\frac{(y - (\mu+\sigma^2))^2}{2\sigma^2} + \frac{\cancel{\mu^2} + 2\mu\sigma^2 + \sigma^4 - \cancel{\mu^2}}{2\sigma^2}\right) dy$$

$$= \exp\left(\mu + \frac{\sigma^2}{2}\right) \frac{1}{\sigma\sqrt{2\pi}} \int_{\ln k}^\infty \exp\left(-\frac{(y - (\mu+\sigma^2))^2}{2\sigma^2}\right) dy$$

$$= \exp\left(\mu + \frac{\sigma^2}{2}\right) P\left(N(\mu+\sigma^2, \sigma^2) > \ln k\right)$$

$$= \exp\left(\mu + \frac{\sigma^2}{2}\right) P\left(N(0,1) > \frac{\ln k - \mu - \sigma^2}{\sigma}\right)$$

$$= \exp\left(\mu + \frac{\sigma^2}{2}\right) \left(1 - \phi\left(\frac{\ln k - \mu}{\sigma} - \sigma\right)\right)$$

$$= \exp\left(\mu + \frac{\sigma^2}{2}\right) \phi\left(\frac{\mu - \ln k}{\sigma} + \sigma\right)$$

(Q. 4)

(b) put option: expiration date T on a forward maturing at time $T' > T \Rightarrow$ payoff $= (K - F(T,T'))^+$

↳ Terminal payoff.

~~So $F(T,T') = F(t,T') e^{(r-\delta-\frac{\sigma^2}{2})(T-t)+\sigma W^q_{(T-t)}}$~~

Premidterm notes → 3.5.1 (Page 30)

Eqn (36):-

$$F(t,T') = S_t e^{(r-\delta)(T'-t)}$$

$$F(T,T') = S_T e^{(r-\delta)(T'-T)}$$

$$\Rightarrow \frac{F(T,T')}{F(t,T')} = \frac{S_T}{S_t} e^{(r-\delta)(t-T)}$$

$$= e^{(r-\delta-\frac{\sigma^2}{2})(T-t)+\sigma(W^q_T-W^q_t)} e^{(r-\delta)(t-T)}$$

$$= e^{-\frac{\sigma^2}{2}(T-t)+\sigma(W^q_T-W^q_t)}$$

$$\Rightarrow \boxed{F(T,T') = F(t,T') e^{-\frac{\sigma^2}{2}(T-t)+\sigma(W^q_T-W^q_t)}}$$

$$\Rightarrow \ln F(T,T') = \ln F(t,T') - \frac{\sigma^2}{2}(T-t) + \sigma W^q_T - W^q_t$$

$$\Rightarrow \boxed{\ln F(T,T') \sim N\left(\ln F(t,T') - \frac{\sigma^2}{2}(T-t), \sigma^2(T-t)\right)}$$

(Q.4)

(c) price $= P_t = e^{-r(T-t)} E\left(\left(K - F(T,T')\right)^+\right)$

$$= e^{-r(T-t)} K\, E\left(\left(1 - \frac{F(T,T')}{K}\right)^+\right)$$

$\downarrow$

lognormal ?

$$\ln\left(\frac{F(T,T')}{K}\right) = \ln\left(\frac{F(t,T')}{K}\, e^{-\frac{\sigma^2}{2}(T-t) + \sigma(W^q_T - W^q_t)}\right)$$

$$= \ln\left(\frac{F(t,T')}{K}\right) - \frac{\sigma^2}{2}(T-t) + \sigma(W^q_T - W^q_t)$$

$$= N\left(\ln\left(\frac{F(t,T')}{K}\right) - \frac{\sigma^2}{2}(T-t),\ \sigma^2(T-t)\right)$$

u_c

$$= N\left(\ln\left(\frac{F(t,T')}{K}\right) - \frac{\sigma^2\Delta}{2},\ \sigma^2\Delta\right) \quad \text{where } \Delta = T-t$$

σ_c^2

$$\Rightarrow P_t = e^{-r\Delta} K\left(\Phi\left(-\frac{u_c}{\sigma_c}\right) - e^{u_c + \sigma_c^2/2}\, \Phi\left(-\frac{u_c}{\sigma_c} - \sigma_c\right)\right)$$

$$= e^{-r\Delta} K\left[\Phi\left(\frac{-\left(\ln\left(\frac{F(t,T')}{K}\right) - \frac{\sigma^2\Delta}{2}\right)}{\sigma\sqrt{\Delta}}\right) - e^{\ln\left(\frac{F(t,T')}{K}\right) - \frac{\sigma^2\Delta}{2} + \frac{\sigma^2\Delta}{2}} \times \Phi\left(\frac{-\left[\ln\left(\frac{F(t,T')}{K}\right) - \frac{\sigma^2\Delta}{2}\right] - \sigma^2\Delta}{\sigma\sqrt{\Delta}}\right)\right]$$

$$= e^{-r\Delta} K\left[\Phi(-d_2) - \frac{F(t,T')}{K}\, \Phi\left(-\frac{\ln\left(\frac{F(t,T')}{K}\right) - \frac{\sigma^2\Delta}{2}}{\sigma\sqrt{\Delta}}\right)\right]$$

$$= e^{-r\Delta} K\, \Phi(-d_2) - e^{-r\Delta} F(t,T')\, \Phi(-d_1)$$

$$= e^{-r\Delta}\left(K\, N(-d_2) - F(t,T')\, N(-d_1)\right)$$

$$= \boxed{e^{-r(T-t)}\left(K\, N(-d_2) - F(t,T')\, N(-d_1)\right)}$$

where $d_1 = \dfrac{\ln\left(\frac{F(t,T')}{K}\right) + \frac{\sigma^2(T-t)}{2}}{\sigma\sqrt{T-t}}$ & $d_2 = d_1 - \sigma\sqrt{T-t}$

(Q.4)(d)

So, $\ln F(t,T') = \ln S_t\, e^{-\alpha(T'-t)} + m(1-e^{-\alpha(T'-t)}) + \frac{\sigma^2}{4\alpha}(1-e^{-2\alpha(T'-t)})$

$\boxed{\text{(Eq}^n \text{ 39 in notes)}}$

$$= Y_t\, e^{-\alpha(T'-t)} + m(1-e^{-\alpha(T'-t)}) + \frac{\sigma^2}{4\alpha}(1-e^{-2\alpha(T'-t)})$$

$$\because Y_t \sim N\Big(\underbrace{Y_0 e^{-\alpha t} + m(1-e^{-\alpha t})}, \underbrace{\frac{\sigma^2}{2\alpha}(1-e^{-2\alpha(t)})}\Big)$$

$$\boxed{\ln(F(t,T')) \sim N\Big(Y_0 e^{-\alpha T'} + m(e^{-\alpha(T'-t)} - e^{-\alpha T'}) + m(1-e^{-\alpha(T'-t)}) + \frac{\sigma^2}{4\alpha}(1-e^{-2\alpha(T'-t)}), \frac{\sigma^2}{2\alpha}(e^{-2\alpha(T'-t)} - e^{-2\alpha T'})\Big)}$$

~~So, $\ln F(t,T') = T_1(t,T')\ln S_t + T_2(t,T') + T_3(t,T')$~~

~~$\ln F(T,T') = T_1(T,T')\ln S_t$~~

where $T_1(a,b) = e^{-\alpha(b-a)}$

$T_2(a,b) = m(1-e^{-\alpha(b-a)})$

$T_3(a,b) = \frac{\sigma^2}{4\alpha}(1-e^{-2\alpha(b-a)})$

~~So, $\ln F(t,T') =$~~

So, $\ln F(T,T') = T_1(T,T')\ln S_T + T_2(T,T') + T_3(T,T')$

$\ln F(t,T') = T_1(t,T')\ln S_t + T_2(t,T') + T_3(t,T')$

$$\Rightarrow \ln F(T,T') - \ln F(t,T') = T_1(T,T')\ln S_T - T_1(t,T')\ln S_t \underbrace{\boxed{+ T_2(T,T') - T_2(t,T') + T_3(T,T') - T_3(t,T')}}_{T_4}$$

$$= e^{-\alpha(T'-T)}\ln S_T - e^{-\alpha(T'-t)}\ln S_t + T_4$$

$$= e^{-\alpha T'}(e^{\alpha T}\ln S_T - e^{\alpha t}\ln S_t) + T_4$$

$$= e^{-\alpha T'}(e^{\alpha T} Y_T - e^{\alpha t} Y_t) + T_4 \quad \to \text{Eq}^n \text{ 37}.$$

$$= e^{-\alpha T'}\Big(e^{\alpha T}\big(Y_t e^{-\alpha(T-t)} + m(1-e^{-\alpha(T-t)}) + \sigma e^{-\alpha T}\int_t^T e^{\alpha u}\, dW_u^{\mathbb{Q}}\big) - e^{\alpha t} Y_t\Big) + T_4$$

$$= e^{-\alpha T'}\Big(\cancel{Y_t e^{\alpha t}} + m(e^{\alpha T} - e^{\alpha t}) + \sigma\int_t^T e^{\alpha u}\, dW_u^{\mathbb{Q}} - \cancel{e^{\alpha t} Y_t}\Big) + T_4$$

$$\cancel{= e^{-\alpha T'}(} = m(e^{-\alpha(T'-T)} - e^{-\alpha(T'-t)}) + T_4 + \sigma\int_t^T e^{-\alpha(T'-u)}\, dW_u^{\mathbb{Q}}$$

$$\therefore T_4 = f^n(T_2, T_3) = \text{deterministic},$$

$$\ln F(T,T') - \ln F(t,T')$$

$$\sim N\left(m\left(e^{-\alpha(T'-T)} - e^{-\alpha(T'-t)}\right) + T_4,\ \sigma^2 \int_t^T e^{-2\alpha(T'-u)}\,du \right)$$

$$= N\left(m\left(e^{-\alpha(T'-T)} - e^{-\alpha(T'-t)}\right) + T_4,\ \frac{\sigma^3}{2\alpha}\left(e^{-2\alpha(T'-T)} - e^{-2\alpha(T'-t)}\right) \right)$$

$$\boxed{\begin{aligned} &\Rightarrow \ln F(T,T') \\ &\sim N\Big(\ln F(t,T') + m\left(e^{-\alpha(T'-T)} - e^{-\alpha(T'-t)}\right) + T_2(T,T') - T_2(t,T') + T_3(T,T') - T_3(t,T'), \\ &\qquad \frac{\sigma^2}{2\alpha}\left(e^{-2\alpha(T'-T)} - e^{-2\alpha(T'-t)}\right) \Big) \end{aligned}}$$

(Q.4)

(e) Let $\ln F(T, T') \sim N(\mu_e, \sigma_e^2)$ → refer to part(d)

$$\Rightarrow P_t = e^{-r(T-t)}(K - F(T,T'))^+$$

$$= e^{-r(T-t)} \cdot K \left(1 - \frac{F(T,T')}{K}\right)^+$$

Now, $\ln\left(\frac{F(T,T')}{K}\right) = \ln F(T,T') - \ln K$

$$= N(\mu_e - \ln K, \sigma_e^2)$$

⇒ Using part(a) of Q. 4,

$$P_t = e^{-r(T-t)} K \left(\Phi\left(\frac{-(\mu_e - \ln K)}{\sigma_e}\right) - e^{(\mu_e - \ln K) + \frac{\sigma_e^2}{2}} \Phi\left(\frac{-(\mu_e - \ln K) - \sigma_e^2}{\sigma_e}\right)\right)$$

$$\Rightarrow \boxed{P_t = e^{-r(T-t)} K\left(\Phi\left(\frac{-\mu_e - \ln K}{\sigma_e}\right) - e^{\mu_e - \ln K + \frac{\sigma_e^2}{2}} \Phi\left(-\frac{(\mu_e - \ln K)}{\sigma_e} - \sigma_e\right)\right)}$$

(Here, $\mu_e = \ln F(t,T') + m(e^{-\alpha(T'-T)} - e^{-\alpha(T'-t)}) + T_2(T,T') - T_2(t,T') + T_3(T,T') - T_3(t,T')$)

& $\sigma_e^2 = \frac{\sigma^2}{2\alpha}(e^{-2\alpha(T'-T)} - e^{-2\alpha(T'-t)})$

⇓

from part (d)

(Q.5) S_t = Stock price

X_t = market Index

$c(S) \xrightarrow[\text{on index } X]{\text{bchmk}}$ $pf = (S_T - X_T)^+$ on date T.

Risk Neutral world : Q

(a) $W_t^{Q1} \perp W_t^{Q2}$ are 2 BM.

Given : $B_t^Q = \rho W_t^{Q1} + \sqrt{1-\rho^2}\, W_t^{Q2}$; $\rho \in (-1,1)$

To show : B_t^Q = BM.

<u>proof</u> :- since W_t^{Q1}, W_t^{Q2} = BM,

$W_t^{Q1} \sim N(0,t)$, $W_t^{Q2} \sim N(0,t)$ (Both W_t^{Q1} & W_t^{Q2} are scalars)

↓

RV

further since $W_t^{Q1} \perp W_t^{Q2}$ $\forall t$,

$corr(W_t^{Q1}, W_t^{Q2}) = 0$

$\Rightarrow cov(W_t^{Q1}, W_t^{Q2}) = 0$

So, B_t^Q = linear comb^n of 2 Ind Gaussians

= Gaussian

$\sim N(0,1)$

$\sim N(0,1)$

$\frac{1}{|\Sigma|^{1/2} 2\pi} e^{-\frac{1}{2}[\,][\,]^{-1}[\,]}$

↓

Not separable in general, but still Gaussian

$E(B_t^Q) = \rho \cdot 0 + \sqrt{1-\rho^2} \cdot 0 = 0$

$var(B_t^Q) = \rho^2 \cdot t + (1-\rho^2)t + \rho\sqrt{1-\rho^2} \cdot 0$

$= t$

$\Rightarrow B_t^Q \sim N(0,t)$ ①

Also, we need to prove some more <u>things</u>.

(Q.) How to prove that something is a Brownian Motion (BM)?

→ ① Stationary & ind^{P} increments : Since W_t^{Q1}, W_t^{Q2} are BM, they have s & i increments ⇒ so would B_t

② $B_0 = 0$: $B_0 = \rho W_0^{Q1} + \sqrt{1-\rho^2}\, W_0^{Q2} = \rho \cdot 0 + \sqrt{1-\rho^2} \cdot 0 = 0$

BM

③ Continuity : Since W_t^{Q1} & W_t^{Q2} are continuous, so will be B_t

④ : $B_t - B_s = \rho\,(W_t^{Q1} - W_s^{Q1}) + \sqrt{1-\rho^2}\,(W_t^{Q2} - W_s^{Q2})$

$= \rho\, N(0, (t-s)) + \sqrt{1-\rho^2}\, N(0, t-s)$

$= N(0, t-s)$ ✓

To prove :-

① $B_t \sim N(0,t)$

② $B_0 = 0$

③ continuity

④ stationary & ind^{P} increments

⑤ $B_t - B_s = N(0, t-s)$

⇒ QED

(b) $E^{Q}(W_T^{Q1} B_T^{Q})$

$= E^{Q}\left(W_T^{Q1}\left(\rho W_T^{Q1} + \sqrt{1-\rho^2}\, W_T^{Q2}\right)\right)$

$= E^{Q}\left(\rho (W_T^{Q1})^2 + \sqrt{1-\rho^2}\, W_T^{Q1} W_T^{Q2}\right)$

$= \rho \cdot (T+0) + \sqrt{1-\rho^2}\, \underbrace{E(W_T^{Q1})}_{0}\, \underbrace{E(W_T^{Q2})}_{0}$

$= \rho T$

(c) No arb price $= e^{-rT} E\left((S_T - X_T)^+\right)$ — ⊛

Now, $\dfrac{dX_t}{X_t} = r\,dt + \eta\, dW_t^{Q1}$

$\Rightarrow X_t = X_0 e^{(r-\eta^2/2)t + \eta W_t^{Q1}}$ — ①

IIIly, $S_t = S_0 e^{(r-\sigma^2/2)t + \sigma B_t^{Q}}$ — ②

So, putting ① & ② in ⊛,

price (P) $= e^{-rT} E\left(\left(S_0 e^{(r-\sigma^2/2)T + \sigma B_T^{Q}} - X_0 e^{(r-\eta^2/2)T + \eta W_T^{Q1}}\right)^+\right)$

$= E\left(\left(S_0 e^{-\sigma^2/2\, T + \sigma B_T^{Q}} - X_0 e^{-\eta^2/2\, T + \eta W_T^{Q1}}\right)^+\right)$

~~$= E\left(S_0 e^{-\frac{\sigma^2}{2}T}\right.$~~

$= E\left(\left(S_0 \exp\left(-\frac{\sigma^2}{2}T + \sigma\left(\rho W_T^{Q1} + \sqrt{1-\rho^2}\, W_T^{Q2}\right)\right) - X_0 \exp\left(-\frac{\eta^2}{2}T + \eta W_T^{Q1}\right)\right)^+\right)$

$e^{-rT} E\left((S_T - X_T)^+\right)$

(d) $C = S_0 N(d_1) - K e^{-rT} N(d_2)$ where $d_1 = \dfrac{\ln\left(\frac{S_0}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}$

$\hookrightarrow$ from $C = E\left(e^{-rT}(S_T - K)^+\right)$

we have:-

$$C = E\left(e^{-rT}(S_T - X_T)^+\right)$$

$$\Rightarrow C = S_0 N(d_1) - X_T e^{-rT} N(d_2)$$

$$\text{or } C = e^{-rT} E^{Q}\left(X_T \left(\frac{S_T}{X_T} - 1\right)^+\right)$$

using change of numeraire,

$$\text{let } \frac{dQ'}{dQ} = X_T = X_0 e^{\left(r - \frac{\eta^2}{2}\right)T + \eta W_T^{Q'}}$$

$$\text{So, } \frac{S_T}{X_T} = \frac{S_0}{X_0} e^{\left(r - \frac{\sigma^2}{2}\right)T + \sigma B_T - \left(r - \frac{\eta^2}{2}\right)T - \eta W_T^{Q'}}$$

$$= \frac{S_0}{X_0} e^{\frac{(\eta^2 - \sigma^2)}{2}T + \sigma B_T - \eta W_T^{Q'}}$$

Now, using Margrabe's formula, $\rightarrow$ Internet

If pf $= (S^1_T - S^2_T)^+ \rightarrow C = e^{-q_1 T} S^1_0 N(d_1) - e^{-q_2 T} S^2_0 N(d_2)$

yield q_1 q_2

vol σ_1 σ_2

corr ρ

where $d_1 = \dfrac{\ln\left(\frac{S^1_0}{S^2_0}\right) + \left(q_2 - q_1 + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}$; $d_2 = d_1 - \sigma\sqrt{T}$

where $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2 - 2\sigma_1\sigma_2\rho}$.

In our case,

$S^1_T = S_T$, $S^2_T = X_T$

yield $r - r = 0$ $r - r = 0$

vol σ η

corr ρ

$$\Rightarrow \boxed{C = \cancel{e^{-rT}}\left(S_0 N(d_1) - X_0 N(d_2)\right)}$$

where $d_1 = \dfrac{\ln\left(\frac{S_0}{X_0}\right) + \frac{\tilde{\sigma}^2}{2}T}{\tilde{\sigma}\sqrt{T}}$; $\tilde{\sigma} = \sqrt{\sigma^2 + \eta^2 - 2\sigma\eta\rho}$

$d_2 = d_1 - \tilde{\sigma}\sqrt{T}$

$\tilde{S}_0 = S_0$, $\tilde{\sigma} = \sqrt{\sigma^2 + \eta^2 - 2\sigma\eta\rho}$

$\tilde{K} = X_0$, $\tilde{r} = 0$

So, if $E^{Q}(X_T H) = E^{Q^x}(H)$,

then $\frac{dQ^x}{dQ} = e^{-rT}\frac{X_T}{X_0} = e^{-r(T-t)}\frac{X_T}{X_t}$

Assumption: Under Q, $e^{-rt}X_t$ is a martingale

$$S_t = S_0 e^{(r-\frac{\sigma^2}{2})t + \sigma B_t}$$

$$E(S_t) = S_0 e^{rt} \Rightarrow E(e^{-rt}S_t) = S_0$$

* Who all are mtgl?
  - $\to f_t$
  - $\to \cancel{S_t}\ e^{-rt}S_t$

Thm:- If X_t is a martingale under Q,

then $E^{Q}(X_T H) = X_0 E^{Q^x}(H)$

where $\frac{dQ^x}{dQ} = \frac{X_T}{X_0}$.

So, $E^{Q}(X_T H) = \int X_T H\, dQ = \cancel{\int X_T H \frac{dQ^x}{dQ}}$

mtgl $\Rightarrow X_T \sim X_0$

$$= \int X_0 H\, dQ^x$$

$$= E^{Q^x}(X_0 H)$$

$$= X_0 E^{Q^x}(H)$$

In our case, $e^{-rT}X_T$ is a martingale under Q.

So, $\bigcirc\!\!\!\!c = e^{-rT}E^Q((S_T - X_T)^+) = e^{-rT}E^Q\left(X_T\left(\frac{S_T}{X_T}-1\right)^+\right)$

price of benchmarked call

$$= E^Q\left((e^{-rT}X_T)\left(\frac{S_T}{X_T}-1\right)^+\right)$$

$$= E^{\tilde{Q}}\left((e^{-r\cdot 0}X_0)\left(\frac{S_T}{X_T}-1\right)^+\right)$$

$$= E^{\tilde{Q}}\left(X_0\left(\frac{S_T}{X_T}-1\right)^+\right)$$

$$= \boxed{X_0\, E^{\tilde{Q}}\left(\left(\frac{S_T}{X_T}-1\right)^+\right)} \quad - (4).$$

Now, when $pf = (S_T - K)^+$, $\frac{dS_t}{S_t} = r\,dt + \sigma\, dW_t$

~~Here~~ $\cancel{\ln\left(\frac{S_T}{X_T}\right) = \ln\left(\frac{S_0}{X_0}\right)}$

So, $\frac{S_t}{X_t} = f(S_t, X_t)$

$$\Rightarrow d\left(\frac{S_t}{X_t}\right) = \cancel{\frac{1}{X_t}dS_t} \; \frac{\partial f}{\partial S_t}dS_t + \frac{\partial f}{\partial X_t}dX_t + \frac{1}{2}\left(\frac{\partial^2 f}{\partial S_t^2}(dS_t)^2 + \frac{\partial^2 f}{\partial X_t^2}(dX_t)^2 + 2\frac{\partial^2 f}{\partial S_t \partial X_t}dS_t\,dX_t\right)$$

$$\Rightarrow d\left(\frac{S_t}{X_t}\right) = \frac{1}{X_t}\cdot dS_t - \frac{S_t}{X_t^2}dX_t + \frac{1}{2}\left(0\cdot(dS_t)^2 + \frac{2S_t}{X_t^3}(dX_t)^2 + 2\frac{(-1)}{X_t^2}dS_t\,dX_t\right)$$

$$= \frac{S_t}{X_t}\left(r\,dt + \sigma\, dB_t^Q\right) - \frac{S_t}{X_t}\left(r\,dt + \eta\, dW_t^{Q1}\right) + \frac{1}{2}\left(\frac{2S_t}{X_t}\left(\frac{dX_t}{X_t}\right)^2 - 2\frac{S_t}{X_t}\left(\frac{dS_t}{S_t}\right)\left(\frac{dX_t}{X_t}\right)\right)$$

$$= \frac{S_t}{X_t}\left(\cancel{r\,dt} + \sigma\, dB_t^Q - \cancel{r\,dt} - \eta\, dW_t^{Q1} + \eta^2 dt - \sigma\eta\delta\, dt\right)$$

$\Rightarrow$ If $\frac{S_t}{X_t} = Z_t$,

$$\frac{dZ_t}{Z_t} = (\eta^2 - \sigma\eta\delta)\,dt + \sigma\, dB_t^Q - \eta\, dW_t^{Q1}$$

$$= (\eta^2 - \sigma\eta\delta)\,dt + (\sigma\delta - \eta)\,W_t^{Q1} + \sigma\sqrt{1-\delta^2}\,W_t^{Q2}$$

$$\Rightarrow \frac{dZ_t}{Z_t} = \eta(\eta-\sigma s)\,dt + \sqrt{(\cancel{\sigma^2 s^2} + \eta^2 - 2\sigma s\eta + \sigma^2 - \cancel{\sigma^2 s^2})}\, dW_t^{new}$$

$$= \boxed{\eta(\eta-\sigma s)}\,dt + \sqrt{\sigma^2+\eta^2-2\sigma\eta s}\,dW_t^{new}$$

$$= \tilde{r}\,dt + \tilde{\sigma}\,dW_t^{new}. \rightarrow$$

this leads to:-

$$Z_T = Z_0 \exp\left(\left(\tilde{r} - \frac{\tilde{\sigma}^2}{2}\right)T + \tilde{\sigma}W_T^{new}\right)$$

$$= Z_0 \exp\left(\left(\eta^2 - \sigma\eta s - \frac{1}{2}(\sigma^2+\eta^2-2\sigma\eta s)\right)T + \tilde{\sigma}W_T^{new}\right)$$

$$= Z_0 \exp\left(\left(\frac{\eta^2-\sigma^2}{2}\right)T + \tilde{\sigma}W_T^{new}\right)$$

So, we have:-

$$c = X_0\, E^{\tilde{Q}}\left((Z_T-1)^+\right)$$

$$= X_0\left(Z_0 N(\tilde{d}_1) - 1\cdot e^{-\tilde{r}T} N(\tilde{d}_2)\right)$$

where $\tilde{d}_1 = \dfrac{\ln(Z_0) + \left(\tilde{r} + \frac{\tilde{\sigma}^2}{2}\right)T}{\tilde{\sigma}\sqrt{T}}$, $\tilde{d}_2 = \tilde{d}_1 - \tilde{\sigma}\sqrt{T}$

$$= X_0\left(\frac{S_0}{X_0} N(\tilde{d}_1) - e^{-\tilde{r}T} N(\tilde{d}_2)\right)$$

where $\tilde{d}_1 = \dfrac{\ln\left(\frac{S_0}{X_0}\right) + \left(\tilde{r} + \frac{\tilde{\sigma}^2}{2}\right)T}{\tilde{\sigma}\sqrt{T}}$, $\tilde{d}_2 = \tilde{d}_1 - \tilde{\sigma}\sqrt{T}$

$$= \boxed{S_0 N(\tilde{d}_1) - X_0 e^{-\tilde{r}T} N(\tilde{d}_2)}$$

$$= \boxed{C_{BS}(S_0, X_0, T, \tilde{r}, \tilde{\sigma})}$$

$$= C_{BS}(\tilde{S}_0, \tilde{K}, T, \tilde{r}, \tilde{\sigma})$$

where $\tilde{S}_0 = S_0$

$\tilde{K} = X_0$

$\tilde{r} = \eta(\eta - \sigma s)$

$\tilde{\sigma} = \sqrt{\sigma^2+\eta^2-2\sigma\eta s}$

Marvin Ertl: 920494256

Parth Laturia: 920360697

Raj Patel: 961146448

HW5_PartII_Vatic_SolarWindLoss

December 8, 2025

1 Part II (Vatic)

[1]:
```
from vatic.engines import Simulator
from vatic.data.loaders import load_input, RtsLoader
from vatic.engines import Simulator

import pandas as pd
import numpy as np

from pathlib import Path
import dill as pickle
import os
from datetime import datetime
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore")
```

[2]:
```
grid_name = "Texas-7k"
RUC_MIPGAPS = {grid_name: 0.01}
SCED_HORIZONS = {grid_name: 4}
grid = grid_name  #For Texas, put 'Texas-7k' or 'Texas-7k_2030'
num_days = 1
init_state_file = None

def get_input_for_simulation(date):
    start_date = date #For Texas pick a date in 2018
    template, gen_data, load_data = load_input(grid, start_date,␣
 ↪num_days=num_days, init_state_file=init_state_file)
    return template, gen_data, load_data, start_date
```

[3]:
```
CO2_data = pd.read_csv("emissions_CO2.csv")
dates = ["2018-01-19", "2018-07-19"]
scenario_index_to_name = {1: "baseline", 2: "cloudy", 3: "wind_equivalent"}
```

1.0.1 Baseline Simulation

[7]:
```
siml = Simulator(template, gen_data, load_data, None,
                 pd.to_datetime(start_date).date(), 1, solver='gurobi',
                 solver_options={}, run_lmps=False, mipgap=RUC_MIPGAPS[grid],
                 load_shed_penalty = 1e4, reserve_shortfall_penalty = 1e3,
                 reserve_factor=0.05, output_detail=3,
                 prescient_sced_forecasts=True, ruc_prescience_hour=0,
                 ruc_execution_hour=16, ruc_every_hours=24,
                 ruc_horizon=48, sced_horizon=SCED_HORIZONS[grid],
                 lmp_shortfall_costs=False,
                 enforce_sced_shutdown_ramprate=False,
                 no_startup_shutdown_curves=False,
                 init_ruc_file=None, verbosity=0,
                 output_max_decimals=4, create_plots=False,
                 renew_costs=None, save_to_csv=False,
                 last_conditions_file=None,)
report_dfs = siml.simulate()
```

```
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-packages/vatic/models/params.py:1022)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:126)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:127)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:130)
Calculating PTDF Matrix Factorization
WARNING: DEPRECATED: The quicksum(linear=…) argument is deprecated and
ignored.  (deprecated in 6.6.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/uc_utils.py:100)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:63)
```

```
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:66)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:58)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:54)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:198)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:199)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:239)
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
```

[8]:
```python
import pickle
with open("report_dfs.pkl", "wb") as f:
    pickle.dump(report_dfs, f)
```

[39]:
```python
thermal_detail = report_dfs["thermal_detail"].reset_index()
thermal_detail_co2 = pd.merge(thermal_detail, co2_data, left_on = "Generator",␣
 ↪right_on = "GEN UID", how = "inner")
# Dispatch assumed to be in MWh
thermal_detail_co2["CO2_Total"] =␣
 ↪thermal_detail_co2["Dispatch"]*thermal_detail_co2["CO2 Emissions Lbs/MWh"]
thermal_detail_co2["CO2_Total"].sum()
```

[39]: np.float64(1063377974.6755733)

1.0.2 Matching solar loss with equivalent wind loss

[4]:
```
def get_actual_energy_columns(df, etype):
    output = []
    for col in df.columns:
        if(etype in col[1] and col[0] == "actl"):
            output.append(col)
    return output

def get_p_value(df):
    solar_cols = get_actual_energy_columns(df, 'Solar')
    wind_cols = get_actual_energy_columns(df, 'Wind')
    solar_sum = sum(df[solar_cols].sum(axis=1))
    wind_sum = sum(df[wind_cols].sum(axis=1))
    print(solar_sum, wind_sum)
    p = (solar_sum * 0.75) / wind_sum
    return p

def change_generation_data(gen_data, scenario):
    if(scenario == 2): # cloudy day
        solar_cols = get_actual_energy_columns(gen_data, "Solar")
        gen_data[solar_cols] *= 0.25
    elif(scenario == 3): # wind equivalend
        wind_cols = get_actual_energy_columns(gen_data, "Wind")
        p = get_p_value(gen_data)
        print("p value", p)
        gen_data[wind_cols] *= (1-p)
    return gen_data

def convert_lbs_to_metric_tons(x):
    '''
    Converts a value in lbs to metric tons
    '''
    return x/2204.6223
```

[5]:
```
for date in dates:
    template, gen_data, load_data, start_date = get_input_for_simulation(date)
    for scenario in [1, 2, 3]:
        gen_data_modified = change_generation_data(gen_data, scenario)
        scenario_name = scenario_index_to_name[scenario]
        print(date, scenario_name, "Started!")
        siml = Simulator(template, gen_data_modified, load_data, None,
                        pd.to_datetime(start_date).date(), 1, solver='gurobi',
                        solver_options={}, run_lmps=False, mipgap=RUC_MIPGAPS[grid],
                        load_shed_penalty = 1e4, reserve_shortfall_penalty = 1e3,
```

```python
                     reserve_factor=0.05, output_detail=3,
                     prescient_sced_forecasts=True, ruc_prescience_hour=0,
                     ruc_execution_hour=16, ruc_every_hours=24,
                     ruc_horizon=48, sced_horizon=SCED_HORIZONS[grid],
                     lmp_shortfall_costs=False,
                     enforce_sced_shutdown_ramprate=False,
                     no_startup_shutdown_curves=False,
                     init_ruc_file=None, verbosity=0,
                     output_max_decimals=4, create_plots=False,
                     renew_costs=None, save_to_csv=False,
                     last_conditions_file=None,)
        # Computation
        report_dfs = siml.simulate()
        with open(f"report_dfs_{date}_{scenario_name}.pkl", "wb") as f:
            pickle.dump(report_dfs, f)
        print(date, scenario_name, "Finished!")
        print("")
```

```
31324.076762651617 505316.56833711837
p value 0.04649176188562154
2018-01-19 wind_equivalent Started!
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-packages/vatic/models/params.py:1022)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:126)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:127)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:130)
Calculating PTDF Matrix Factorization
WARNING: DEPRECATED: The quicksum(linear=…) argument is deprecated and
ignored.  (deprecated in 6.6.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/uc_utils.py:100)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
```

```
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:63)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:66)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:58)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/power_vars.py:54)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:198)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:199)
WARNING: DEPRECATED: Using __getitem__ to return a set value from its
(ordered) position is deprecated.  Please use at()  (deprecated in 6.1, will
be removed in (or after) 7.0) (called from /home/pl7830/.conda/envs/vatic-
test/lib/python3.11/site-
packages/egret/model_library/unit_commitment/startup_costs.py:239)
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
2018-01-19 wind_equivalent Finished!

50785.63020434008 391233.68739672063
p value 0.09735670490622052
2018-07-19 wind_equivalent Started!
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
```

```
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Warning for adding constraints: zero or small (< 1e-13) coefficients, ignored
Truncating shutdown_curve longer than scaled minimum down time 1 for generator
50304_AllOther_GEN2
Truncating shutdown_curve longer than scaled minimum down time 1 for generator
50304_AllOther_GEN2
Truncating shutdown_curve longer than scaled minimum down time 1 for generator
50304_AllOther_GEN2
Truncating shutdown_curve longer than scaled minimum down time 1 for generator
50304_AllOther_GEN2
2018-07-19 wind_equivalent Finished!
```

[5]:

```
# Computing CO2 Emissions
CO2_df = pd.DataFrame(columns = ["date", "scenario_name", "CO2 Emissions (in␣
↪MT)"])
row = 0
for date in dates:
    for scenario in [1, 2, 3]:
        scenario_name = scenario_index_to_name[scenario]
        # Fetching the stored pickle files
        with open(f"report_dfs_{date}_{scenario_name}.pkl", "rb") as f:
            report_dfs = pickle.load(f)
        # Extracting thermal details
        thermal_detail = report_dfs["thermal_detail"].reset_index()
        thermal_detail_CO2 = pd.merge(thermal_detail, CO2_data, left_on =␣
↪"Generator", right_on = "GEN UID", how = "inner")
        # Dispatch assumed to be in MWh
        thermal_detail_CO2["CO2_Total"] =␣
↪thermal_detail_CO2["Dispatch"]*thermal_detail_CO2["CO2 Emissions Lbs/MWh"]
        CO2_total = thermal_detail_CO2["CO2_Total"].sum()
        CO2_total = convert_lbs_to_metric_tons(CO2_total)
        print(date, scenario_name, CO2_total)
        CO2_df.loc[row] = [date, scenario_name, CO2_total]
        row += 1
        print("")
```

```
2018-01-19 baseline 482340.20615484717

2018-01-19 cloudy 486732.5932620959

2018-01-19 wind_equivalent 487034.04493126826
```

```
2018-07-19 baseline 677450.4814349589

2018-07-19 cloudy 684518.494041032

2018-07-19 wind_equivalent 686277.0734749862
```

[6]:
```python
# temp = CO2_df.set_index(["date", "scenario_name"])
CO2_df["base_emissions"] = 0
for date in dates:
    val = CO2_df[(CO2_df["date"] == date) & (CO2_df["scenario_name"] ==␣
 ↪"baseline")]["CO2 Emissions (in MT)"].iloc[0]
    CO2_df.loc[CO2_df["date"] == date, "base_emissions"] = val
CO2_df["pct_change"] = (CO2_df["CO2 Emissions (in MT)"] -␣
 ↪CO2_df["base_emissions"])*100/CO2_df["base_emissions"]
CO2_df
```

[6]:
```
         date    scenario_name  CO2 Emissions (in MT)  base_emissions  \
0  2018-01-19         baseline          482340.206155   482340.206155
1  2018-01-19           cloudy          486732.593262   482340.206155
2  2018-01-19  wind_equivalent          487034.044931   482340.206155
3  2018-07-19         baseline          677450.481435   677450.481435
4  2018-07-19           cloudy          684518.494041   677450.481435
5  2018-07-19  wind_equivalent          686277.073475   677450.481435

   pct_change
0    0.000000
1    0.910641
2    0.973139
3    0.000000
4    1.043325
5    1.302913
```

As seen, CO2 emissions increase in both cases when the solar or the wind generation is curtailed. Further, the increase is more in case of wind curtailment (0.97% vs 0.91%, 1.30% vs 1.04%) as compared to solar curtailment, indicating that wind saves more CO2 from being emitted while fulfilling the demand, i.e. while not being in a shortfall shock situation. (Note that generation was only changed on the "actual" columns.) Since the wind and sun shortfall are matched in MWh space summed across the day, the fact that the corresponding pct change in CO2 emissions is higher for wind could be driven by, for instance, the fact that covering a solar generation shortfall is less carbon intense than covering a wind generation shortfall of the same energy amount due to the different hours of the day in which they would occur. A shortfall in solar would affect the grid predominantly while the sun is shining, and during these hours the backup generation in case of a shock may be less carbon-intensive. A wind shortfall, in contrast would affect the grid uniformly throught the day and night, and

at night the backup generation to cover a negative shock in wind may be more carbon intensive.

HW5_20251208_222500_demand_shock

December 8, 2025

1 Part II (Vatic)

[1]:
```
from vatic.engines import Simulator
from vatic.data.loaders import load_input, RtsLoader
from vatic.engines import Simulator

import pandas as pd
import numpy as np

from pathlib import Path
import dill as pickle
import os
from datetime import datetime
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore")
```

[61]:
```
import datetime
import matplotlib.pyplot as plt
import pickle
```

[3]:
```
grid_name = "Texas-7k"
RUC_MIPGAPS = {grid_name: 0.01}
SCED_HORIZONS = {grid_name: 4}
grid = grid_name  #For Texas, put 'Texas-7k' or 'Texas-7k_2030'
num_days = 1
init_state_file = None
```

[63]:
```
def get_input_for_simulation(date):
    '''
    Returns the files, objects to serve as an input to the simulator
    Args:
        date (str): the date for which to fetch the input data
    Returns:
        template, gen_data, load_data, start_date
    '''
```

```python
    start_date = date #For Texas pick a date in 2018
    template, gen_data, load_data = load_input(grid, start_date,␣
 ↪num_days=num_days, init_state_file=init_state_file)
    return template, gen_data, load_data, start_date

def get_actual_energy_columns(df, etype):
    '''
    Returns the column names of energy: etype that are actual columns
    Args:
        df (pd.DataFrame): the dataframe containing columns to be filtered
        etype (string): the type of energy source required
    Returns:
        a list of required column names (list[str])
    '''
    output = []
    for col in df.columns:
        if(etype in col[1] and col[0] == "actl"):
            output.append(col)
    return output

def get_p_value(df):
    '''
    Computes the p-value for which change in wind energy equals 75% reduction␣
 ↪of solar energy
    Args:
        df (pd.DataFrame): the dataframe containing wind and solar data
    Returns:
        p (float): the p-value for which change in wind energy equals 75%␣
 ↪change in solar energy
    '''
    solar_cols = get_actual_energy_columns(df, 'Solar')
    wind_cols = get_actual_energy_columns(df, 'Wind')
    solar_sum = sum(df[solar_cols].sum(axis=1))
    wind_sum = sum(df[wind_cols].sum(axis=1))
    print(solar_sum, wind_sum)
    p = (solar_sum * 0.75) / wind_sum
    return p

def change_generation_data(gen_data, scenario):
    '''
    Changes the generation data based on the scenario to be studied
    Args:
        gen_data (pd.DataFrame): dataframe containing the generation data
        scenario (int, in [1, 2, 3]): scenario number (must be 1 or 2 or 3)
    Returns:
        the generation data modified as per scenario (pd.DataFrame)
    '''
```

```
    if(scenario == 2): # cloudy day
        solar_cols = get_actual_energy_columns(gen_data, "Solar")
        gen_data[solar_cols] *= 0.25
    elif(scenario == 3): # wind equivalend
        wind_cols = get_actual_energy_columns(gen_data, "Wind")
        p = get_p_value(gen_data)
        print("p value", p)
        gen_data[wind_cols] *= (1-p)
    return gen_data

def convert_lbs_to_metric_tons(x):
    '''
    Converts a value in lbs to metric tons
    Args:
        x (float): quantity in lbs
    Returns:
        quantity in metric tons (float)
    '''
    return x/2204.6223

def change_load_data(load_data):
    '''
    Increases the demand (through load_data) between 10:00 to 18:00 hours by 40%
    Args:
        load_data (pd.DataFrame): load data
    Returns:
        modified load data (pd.DataFrame)
    '''
    load_data_input = load_data.copy()
    v = load_data_input.index
    # Filtering on the July date
    v2 = v[v.date == pd.to_datetime("2018-07-19").date()]
    # Filtering on the times to be updated
    v3 = v2[(v2.time >= datetime.time(10)) & (v2.time <= datetime.time(18))]
    load_data_input.loc[v3, "actl"] = 1.4 * load_data_input.loc[v3, "actl"].
↪values
    return load_data_input

def get_total_CO2_emitted(report_dfs):
    '''
    Computes the total CO2 emitted during the entire day (using thermal detail␣
↪dataframe in report dfs and CO2 emissions data for each generator)
    Args:
        report_dfs (dict[str, pd.DataFrame]: the master data object
    Returns:
        the total CO2 emitted (float)
    '''
```

```
    # Extracting thermal details
    thermal_detail = report_dfs["thermal_detail"].reset_index()
    thermal_detail_CO2 = pd.merge(thermal_detail, CO2_data, left_on =␣
 ↪"Generator", right_on = "GEN UID", how = "inner")
    # Dispatch assumed to be in MWh
    thermal_detail_CO2["CO2_Total"] =␣
 ↪thermal_detail_CO2["Dispatch"]*thermal_detail_CO2["CO2 Emissions Lbs/MWh"]
    CO2_total = thermal_detail_CO2["CO2_Total"].sum()
    CO2_total = convert_lbs_to_metric_tons(CO2_total)
    return CO2_total
```

[5]:

```
CO2_data = pd.read_csv("emissions_CO2.csv")
dates = ["2018-01-19", "2018-07-19"]
scenario_index_to_name = {1: "baseline", 2: "cloudy", 3: "wind_equivalent"}
```

1.0.1 Summer Daytime Demand Shock

[ ]:

```
date = "2018-07-19"
template, gen_data, load_data, start_date = get_input_for_simulation(date)
load_data_modified = change_load_data(load_data)
siml = Simulator(template, gen_data, load_data_modified, None,
            pd.to_datetime(start_date).date(), 1, solver='gurobi',
            solver_options={}, run_lmps=False, mipgap=RUC_MIPGAPS[grid],
            load_shed_penalty = 1e4, reserve_shortfall_penalty = 1e3,
            reserve_factor=0.05, output_detail=3,
            prescient_sced_forecasts=True, ruc_prescience_hour=0,
            ruc_execution_hour=16, ruc_every_hours=24,
            ruc_horizon=48, sced_horizon=SCED_HORIZONS[grid],
            lmp_shortfall_costs=False,
            enforce_sced_shutdown_ramprate=False,
            no_startup_shutdown_curves=False,
            init_ruc_file=None, verbosity=0,
            output_max_decimals=4, create_plots=False,
            renew_costs=None, save_to_csv=False,
            last_conditions_file=None,)
report_dfs = siml.simulate()

# Storing output
with open(f"report_dfs_{date}_demand_shock.pkl", "wb") as f:
    pickle.dump(report_dfs, f)
```

[23]:

```
with open(f"report_dfs_{date}_demand_shock.pkl", "rb") as f:
    report_dfs_ds = pickle.load(f)

with open(f"report_dfs_{date}_baseline.pkl", "rb") as f:
    report_dfs_bl = pickle.load(f)
```

CO2 Emission Analysis

[24]:
```
co2_baseline = get_total_CO2_emitted(report_dfs_bl)
co2_shock = get_total_CO2_emitted(report_dfs_ds)
print("CO2 Emission during demand shock:", co2_shock)
print("CO2 Emission during baseline:", co2_baseline)
```

```
CO2 Emission during demand shock: 719156.190608769
CO2 Emission during baseline: 677450.4814349589
```

Total Thermal Dispatch, by Technology

[45]:
```
print("Under Demand Shock:")
data_ds = report_dfs_ds["thermal_detail"].copy()
data_ds["Tech"] = data_ds.reset_index()["Generator"].str.split("_").
 ↪transform(lambda x: x[1]).values
data_ds.groupby("Tech")["Dispatch"].sum().sort_values(ascending = False).
 ↪round(2)
```

```
Under Demand Shock:
```

[45]:
```
Tech
NaturalGasFiredCombinedCycle          623610.25
ConventionalSteamCoal                 284203.92
NaturalGasSteamTurbine                152954.22
Nuclear                               111339.38
NaturalGasFiredCombustionTurbine       65581.39
NaturalGasInternalCombustionEngine      3525.32
OtherGases                              3221.43
AllOther                                1871.23
Wood/WoodWasteBiomass                   1287.45
Name: Dispatch, dtype: float64
```

[46]:
```
print("For Baseline:")
data_bl = report_dfs_bl["thermal_detail"].copy()
data_bl["Tech"] = data_bl.reset_index()["Generator"].str.split("_").
 ↪transform(lambda x: x[1]).values
data_bl.groupby("Tech")["Dispatch"].sum().sort_values(ascending = False).
 ↪round(2)
```

```
For Baseline:
```

[46]:
```
Tech
NaturalGasFiredCombinedCycle          619197.45
ConventionalSteamCoal                 265605.25
NaturalGasSteamTurbine                124505.32
Nuclear                               106637.38
NaturalGasFiredCombustionTurbine       61762.96
NaturalGasInternalCombustionEngine      3395.75
OtherGases                              3221.43
```

```
AllOther                                 1778.14
Wood/WoodWasteBiomass                    1249.96
Name: Dispatch, dtype: float64
```

Headroom Analysis, by Tech

[47]:
```python
print("Under Demand Shock:")
data_ds.groupby("Tech")["Headroom"].sum().sort_values(ascending = False).
 ↪round(2)
```

```
Under Demand Shock:
```

[47]:
```
Tech
NaturalGasFiredCombinedCycle           41944.03
NaturalGasSteamTurbine                 24872.46
ConventionalSteamCoal                   3947.69
NaturalGasFiredCombustionTurbine        3218.16
AllOther                                 185.02
NaturalGasInternalCombustionEngine       183.52
Wood/WoodWasteBiomass                    175.98
OtherGases                                -0.00
Nuclear                                   -0.00
Name: Headroom, dtype: float64
```

[48]:
```python
print("For Baseline:")
data_bl.groupby("Tech")["Headroom"].sum().sort_values(ascending = False).
 ↪round(2)
```

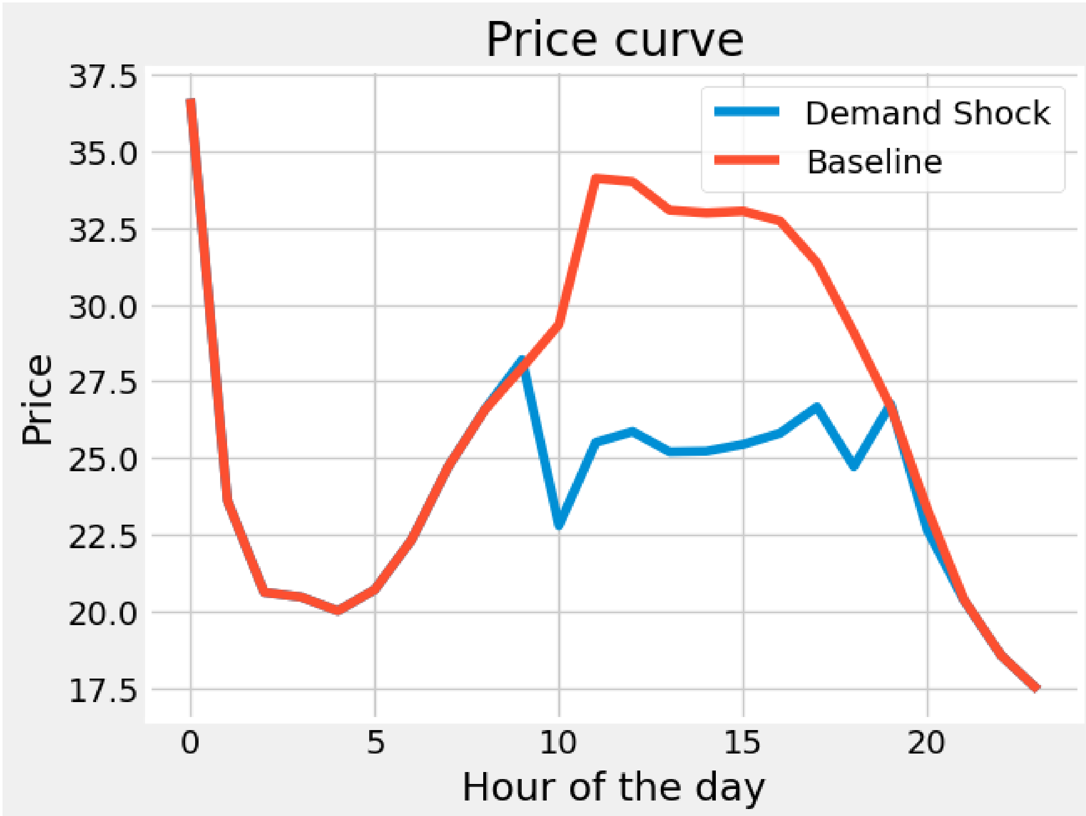
```
For Baseline:
```

[48]:
```
Tech
NaturalGasFiredCombinedCycle           43836.01
NaturalGasSteamTurbine                 39132.30
ConventionalSteamCoal                  11278.96
NaturalGasFiredCombustionTurbine        6059.70
Nuclear                                 2625.64
NaturalGasInternalCombustionEngine       306.84
AllOther                                 245.89
Wood/WoodWasteBiomass                    213.47
OtherGases                                -0.00
Name: Headroom, dtype: float64
```

Price Analysis

[62]:
```python
plt.plot(report_dfs_ds['hourly_summary']['Price'].values, label = "Demand␣
 ↪Shock")
plt.plot(report_dfs_bl['hourly_summary']['Price'].values, label = "Baseline")
plt.legend()
plt.xlabel("Hour of the day")
plt.ylabel("Price")
```

```
plt.title("Price curve")
plt.show()
```



Observations

- Dispatch is increasing under the demand shock (which is expected since the higher the demand, the more dispatch that has to take place); Also note that dispatch increased across all tech (except OtherGases)
- As expected, the headroom decreased under the demand shock since a part of the original headroom had to be used to fulfil the demand shock!
- Nuclear has a lot of headroom in baseline scenario but falls during the demand shock!
- The relative order of the technologies, (ordered by their dispatch or their headroom) remains more or less the same irrespective of whether the scenario is a baseline or that of a demand shock!
- Surprisingly, the electricity price fell even though the load was increased by 40% between 10:00 and 18:00 on July 19, 2018. By comparing the dispatch and headroom results in the baseline and shocked cases, we observe that the shock changed the set of generators that were dispatched. The new mix might have brought cheaper units to the margin, which determines the market price. In particular, natural gas steam turbines ramped up significantly, with dispatch increasing by about 28,449 MWh. We also see that nuclear generation was pushed to its full capacity after the demand shock, and these units typically have lower costs.

A5_Intensity

December 8, 2025

0.0.1 Question 2

[ ]:

```

```

[9]:

```
import numpy as np
from scipy.sparse import eye, kron, lil_matrix
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import pandas as pd

# SECTION 1: HJB SOLVER

def price(s, x, s_eps=1e-12):
    """
    Price function P(s, x) = sqrt(x / max(s, s_eps)) - 1.
    """
    return np.sqrt(x / np.maximum(s, s_eps)) - 1.0


def build_dx_operator(J, I):
    """
    Build sparse Ax operator corresponding to forward difference in x.
    """
    Dx = lil_matrix((J + 1, J + 1))
    for jj in range(J):
        Dx[jj, jj] = -1.0
        Dx[jj, jj + 1] = 1.0
    Dx = Dx.tocsr()

    Is = eye(I + 1, format="csr")
    Ax = kron(Dx, Is, format="csr")
    return Ax


def build_ds_operators(delta_vec, ds, I, J):
    """
    Build the list of As operators, one for each technology.
```

```
    """
    dtech = len(delta_vec)
    As_list = []

    for t in range(dtech):
        delta = delta_vec[t]
        k = int(round(delta / ds))  # stride in s
        Ds = lil_matrix((I + 1, I + 1))

        if k > 0:
            # forward difference in s
            last_row = I - k + 1
            if last_row >= 1:
                for ii in range(last_row):
                    Ds[ii, ii] = -1.0
                    Ds[ii, ii + k] = 1.0
        elif k < 0:
            # backward difference in s (for negative delta)
            kk = -k
            first_row = kk
            if first_row <= I:
                for ii in range(first_row, I + 1):
                    Ds[ii, ii] = -1.0
                    Ds[ii, ii + k] = 1.0
        else:
            # k == 0 (no change in s) - nothing to do
            pass

        Ds = Ds.tocsr()
        Ix = eye(J + 1, format="csr")
        As = kron(Ix, Ds, format="csr")
        As_list.append(As)

    return As_list


def multi_rhs(t, V, As_list, Ax, Rvec, r, lambdaX, beta_vec, rho_vec, expo_vec):
    """
    Right-hand side of the HJB ODE system dV/dt = ...
    """
    V = V.reshape(-1, 1)
    Ntech = len(As_list)
    Phi_sum = np.zeros_like(V)

    for j in range(Ntech):
        AsV = As_list[j].dot(V)          # discrete Δ^s_j V
        tmp = AsV - rho_vec[j]
```

```
        # Optimal  _j: (Δ^s_j V -  _j)^{1/( _j - 1)}_+
        lam = np.zeros_like(tmp)
        mask = tmp > 0.0
        lam[mask] = np.power(tmp[mask], expo_vec[j])

        Phi_j = np.zeros_like(tmp)
        if np.any(mask):
            lam_pos = lam[mask]
            z_pos = AsV[mask]
            beta_j = beta_vec[j]

            # Hamiltonian term:   z - C( ), with C( ) =    +  ^ /
            Phi_j[mask] = (
                lam_pos * z_pos
                - (1.0 / beta_j) * np.power(lam_pos, beta_j)
                - rho_vec[j] * lam_pos
            )

        Phi_sum += Phi_j

    # HJB ODE in time, backward direction
    dV = -Phi_sum - lambdaX * (Ax.dot(V)) - Rvec.reshape(-1, 1) + r * V
    return dV.ravel()


def solve_hjb(
    T,
    r,
    lambdaX,
    dx,
    ds,
    s_min,
    I,
    J,
    delta_vec,
    beta_vec,
    rho_vec,
    tech_names=None,
    s_eps=1e-12,
    verbose=True,
):
    """
    Solve the HJB equation and return V(0,s,x) and  *_j(0,s,x).
    """
    if verbose:
        print(f"Solving HJB: T={T}, r={r},  _X={lambdaX}")
```

```
    print(f"Grid: I={I}, J={J}, ds={ds}, dx={dx}")

# Grids
S = s_min + ds * np.arange(I + 1)
X = dx * np.arange(J + 1)
Sgrid, Xgrid = np.meshgrid(S, X, indexing="ij")

dtech = len(delta_vec)
if tech_names is None:
    tech_names = [f"Tech_{i+1}" for i in range(dtech)]

# Sanity checks
k_stride = np.round(delta_vec / ds).astype(int)
if np.any(np.abs(k_stride * ds - delta_vec) > 1e-12):
    raise ValueError("Each delta_vec[j] must be integer multiple of ds.")
if np.any(beta_vec <= 1.0):
    raise ValueError("All beta_vec[j] must be > 1.")

# exponent 1/( -1) used in  *
expo_vec = 1.0 / (beta_vec - 1.0)

# Instantaneous reward
Rmat = Sgrid * price(Sgrid, Xgrid, s_eps=s_eps)
Rvec = Rmat.reshape(-1, order="F")
N = Rvec.size

# Operators
Ax = build_dx_operator(J, I)
As_list = build_ds_operators(delta_vec, ds, I, J)

# RHS for ODE solver
def rhs(t, V):
    return multi_rhs(
        t, V, As_list, Ax, Rvec, r, lambdaX, beta_vec, rho_vec, expo_vec
    )

# Terminal condition: V(T) = 0
V_T = np.zeros(N)

if verbose:
    print("Integrating HJB backward in time...")
sol = solve_ivp(
    rhs,
    t_span=(T, 0.0),
    y0=V_T,
    method="RK45",
    rtol=1e-6,
```

```
        atol=1e-9,
    )

    if not sol.success:
        raise RuntimeError(f"ODE solve failed: {sol.message}")
    if verbose:
        print("Integration successful")

    # V(0)
    V0_vec = sol.y[:, -1]
    V0 = V0_vec.reshape((I + 1, J + 1), order="F")

    # Compute  *_j(0,s,x) from V0
    LambdaStar0_j = np.zeros((I + 1, J + 1, dtech))

    for idx_tech in range(dtech):
        k = k_stride[idx_tech]
        Dsv0 = np.zeros_like(V0)

        # finite differences in s depending on stride sign
        if k > 0:
            last_row = I - k + 1
            if last_row >= 1:
                Dsv0[0:last_row, :] = V0[k:k + last_row, :] - V0[0:last_row, :]
        elif k < 0:
            kk = -k
            first_row = kk
            if first_row <= I:
                Dsv0[first_row:I + 1, :] = (
                    V0[0:I + 1 - kk, :] - V0[first_row:I + 1, :]
                )

        tmp = Dsv0 - rho_vec[idx_tech]
        lam = np.zeros_like(tmp)
        mask = tmp > 0.0
        lam[mask] = np.power(tmp[mask], expo_vec[idx_tech])

        LambdaStar0_j[:, :, idx_tech] = lam

    return {
        "S": S,
        "X": X,
        "Sgrid": Sgrid,
        "Xgrid": Xgrid,
        "V0": V0,
        "LambdaStar0_j": LambdaStar0_j,
        "tech_names": tech_names,
```

```python
    }


# SECTION 2: Questions

def run_part2():
    # Base Case Parameters
    T = 5.0
    r = 0.02
    lambdaX = 4.0
    I, J = 50, 50
    s_min = 0.1
    ds, dx = 1.0, 1.0

    # Technologies A, B, C, D
    delta_vec = np.array([3.0, 2.0, 5.0, -1.0])
    beta_vec = np.array([2.0, 2.0, 2.0, 2.0])
    rho_vec = np.array([0.2, 0.2, 0.2, 0.2])
    tech_names = ["A", "B", "C", "D"]

    # Solve Base Case
    res = solve_hjb(
        T, r, lambdaX, dx, ds, s_min, I, J,
        delta_vec, beta_vec, rho_vec, tech_names
    )

    S, X, V0 = res["S"], res["X"], res["V0"]
    Sgrid, Xgrid = res["Sgrid"], res["Xgrid"]
    L_star = res["LambdaStar0_j"]


    # Question 2: Value function surface + slices

    # 3D surface plot of V(0,s,x)
    fig = plt.figure(figsize=(10, 6))
    ax = fig.add_subplot(111, projection="3d")
    surf = ax.plot_surface(Sgrid, Xgrid, V0, edgecolor="none", cmap='viridis')
    ax.set_title("Value Function V(0, s, x)")
    ax.set_xlabel("Supply s")
    ax.set_ylabel("Demand x")
    ax.set_zlabel("V(0,s,x)")
    fig.colorbar(surf, shrink=0.5, aspect=5)
    plt.tight_layout()
    plt.show()

    # 3b. Slices: V vs s for fixed x (for Q2a + curvature in s)
    idx_x10 = np.abs(X - 10).argmin()
```

```
idx_x30 = np.abs(X - 30).argmin()
x_val_10 = X[idx_x10]
x_val_30 = X[idx_x30]

V_slice_x10 = V0[:, idx_x10]   # all s, x  10
V_slice_x30 = V0[:, idx_x30]   # all s, x  30

plt.figure(figsize=(7, 4))
plt.plot(S, V_slice_x10, label=fr"x  {x_val_10:.1f}")
plt.plot(S, V_slice_x30, label=fr"x  {x_val_30:.1f}")
plt.xlabel("Capacity s")
plt.ylabel(r"V(0,s,x)")
plt.title("Slices of V(0,s,x) vs s (fixed x)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# 3c. Slices: V vs x for fixed s (for Q2b + curvature in x)
idx_s10 = np.abs(S - 10).argmin()
idx_s20 = np.abs(S - 20).argmin()
s_val_10 = S[idx_s10]
s_val_20 = S[idx_s20]

V_slice_s10 = V0[idx_s10, :]   # all x, s  10
V_slice_s20 = V0[idx_s20, :]   # all x, s  20

plt.figure(figsize=(7, 4))
plt.plot(X, V_slice_s10, label=fr"s  {s_val_10:.1f}")
plt.plot(X, V_slice_s20, label=fr"s  {s_val_20:.1f}")
plt.xlabel("Demand x")
plt.ylabel(r"V(0,s,x)")
plt.title("Slices of V(0,s,x) vs x (fixed s)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# Question 3: Cross-sections of optimal intensities  *_j

fig, axes = plt.subplots(1, 2, figsize=(14, 5))

# x  10
ax_l = axes[0]
for i, name in enumerate(tech_names):
    ax_l.plot(S, L_star[:, idx_x10, i], label=f"Tech {name}")
ax_l.set_title(fr"Optimal Intensities at x  {x_val_10:.1f}")
```

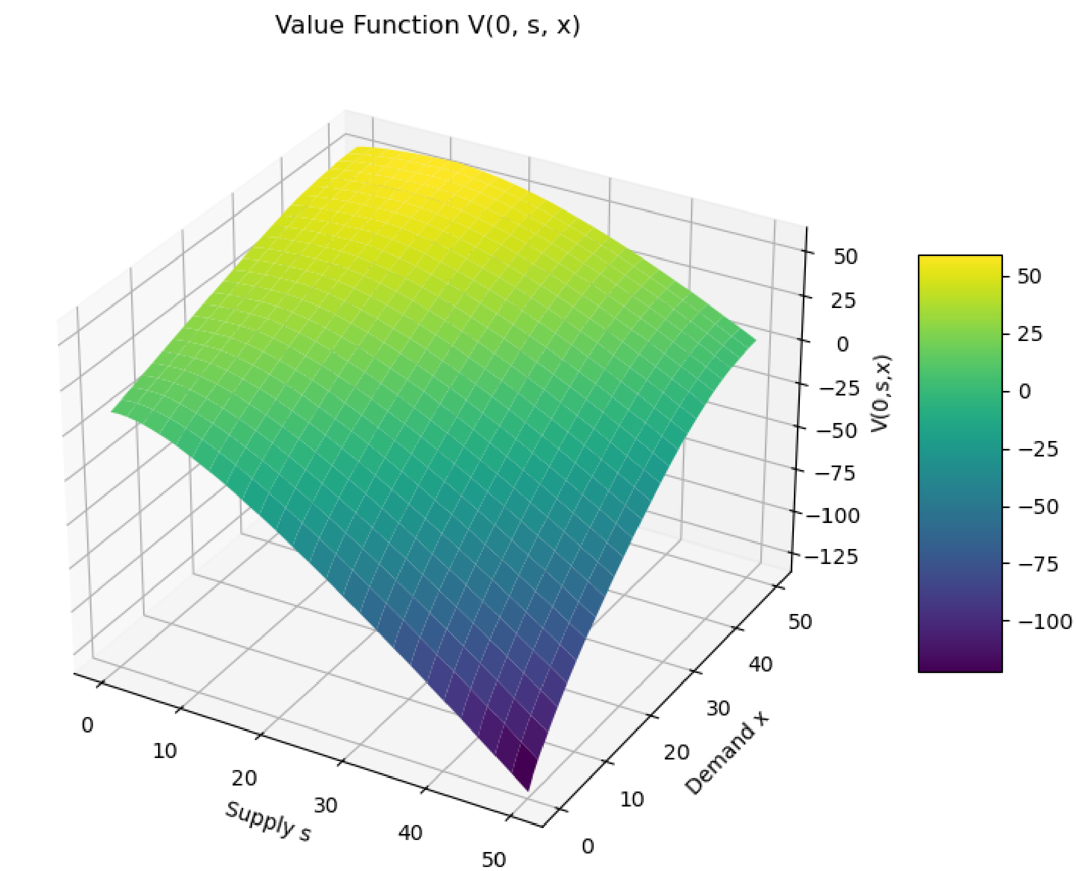
```
ax_l.set_xlabel("Capacity s")
ax_l.set_ylabel(r"Intensity $\lambda_j^*(0,s,x)$")
ax_l.legend()
ax_l.grid(True)

# x   30
ax_r = axes[1]
for i, name in enumerate(tech_names):
    ax_r.plot(S, L_star[:, idx_x30, i], label=f"Tech {name}")
ax_r.set_title(fr"Optimal Intensities at x   {x_val_30:.1f}")
ax_r.set_xlabel("Capacity s")
ax_r.set_ylabel(r"Intensity $\lambda_j^*(0,s,x)$")
ax_r.legend()
ax_r.grid(True)

plt.tight_layout()
plt.show()

# Policy Implications & Subsidy Design

print("\n--- Policy Analysis ---")

# Define scenarios for Technology A
scenarios = {
    "Baseline":       {"rho_A": 0.2,  "beta_A": 2.0},
    "Subsidize rho":  {"rho_A": 0.1,  "beta_A": 2.0},
    "Subsidize beta": {"rho_A": 0.2,  "beta_A": 1.6},
    "Subsidize both": {"rho_A": 0.15, "beta_A": 1.8},
}

results_policy = {}

# Solve HJB under each policy scenario
for name, params in scenarios.items():
    r_vec_new = rho_vec.copy()
    b_vec_new = beta_vec.copy()
    r_vec_new[0] = params["rho_A"]   # Tech A index 0
    b_vec_new[0] = params["beta_A"]

    print(f"Solving scenario: {name}")
    sol = solve_hjb(
        T, r, lambdaX, dx, ds, s_min, I, J,
        delta_vec, b_vec_new, r_vec_new, tech_names,
        verbose=False
    )
    results_policy[name] = sol
```

```python
#  5a. 3D surfaces of  *_A(0,s,x) for each scenario
for name, sol in results_policy.items():
    lamA = sol["LambdaStar0_j"][:, :, 0]  # Tech A
    Sg = sol["Sgrid"]
    Xg = sol["Xgrid"]

    fig = plt.figure(figsize=(9, 6))
    ax3d = fig.add_subplot(111, projection="3d")
    surf = ax3d.plot_surface(Sg, Xg, lamA, edgecolor="none", cmap='viridis')
    ax3d.set_title(rf"Tech A intensity $\lambda_A^*(0,s,x)$ - {name}")
    ax3d.set_xlabel("Supply s")
    ax3d.set_ylabel("Demand x")
    ax3d.set_zlabel(r"$\lambda_A^*$")
    fig.colorbar(surf, shrink=0.5, aspect=5)
    plt.tight_layout()
    plt.show()

# 5b. Cross-section of  *_A vs s at x   30 for all scenarios
fig, ax = plt.subplots(figsize=(10, 6))
for name, sol in results_policy.items():
    lam_A = sol["LambdaStar0_j"][:, idx_x30, 0]  # Tech A
    ax.plot(S, lam_A, label=name)

ax.set_title(fr"Tech A intensity $\lambda_A^*(0,s,x)$ at x   {x_val_30:.
↪1f}")
ax.set_xlabel("Capacity s")
ax.set_ylabel(r"Intensity $\lambda_A^*$")
ax.legend()
ax.grid(True)
plt.tight_layout()
plt.show()

# 5c. Pointwise effectiveness table at a selected state (s,x)
idx_s_test = np.abs(S - 5).argmin()
idx_x_test = idx_x30

table_data = []
base_val = results_policy["Baseline"]["LambdaStar0_j"][idx_s_test,␣
↪idx_x_test, 0]

for name, sol in results_policy.items():
    val = sol["LambdaStar0_j"][idx_s_test, idx_x_test, 0]
    if abs(base_val) > 1e-8:
        pct_change = (val - base_val) / base_val * 100.0
        pct_str = f"{pct_change:.1f}%"
    else:
        pct_str = "NA"
```

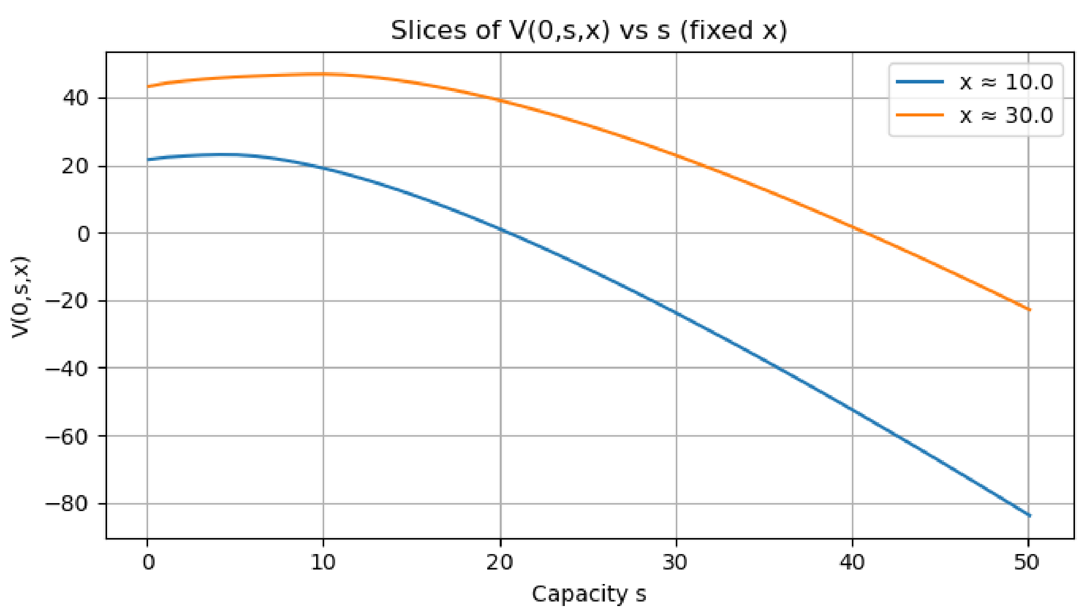
```
        table_data.append(
            {
                "Scenario": name,
                "Lambda_A": f"{val:.4f}",
                "Abs Change": f"{val - base_val:.4f}",
                "% Change": pct_str,
            }
        )

    df = pd.DataFrame(table_data)
    print(
        f"\nPolicy effectiveness at s={S[idx_s_test]:.1f}, x={X[idx_x_test]:.
↪1f}"
    )
    print(df.to_string(index=False))

    # 5c. Pointwise effectiveness table at selected states
    # We compare three distinct regimes:
    # 1. Early Aggressive Growth (s=1.0): Expect lambda > 1
    # 2. Moderate Expansion (s=5.0): Expect lambda < 1
    # 3. Near Saturation (s=9.0): Expect lambda to be small
    test_points = [
        {"s": 1.0, "x": 30.0},
        {"s": 5.0, "x": 30.0},
        {"s": 9.0, "x": 30.0}
    ]

    all_table_data = []

    for pt in test_points:
        s_val = pt["s"]
        x_val = pt["x"]

        idx_s_test = np.abs(S - s_val).argmin()
        idx_x_test = np.abs(X - x_val).argmin()

        base_val = results_policy["Baseline"]["LambdaStar0_j"][idx_s_test,␣
↪idx_x_test, 0]

        for name, sol in results_policy.items():
            val = sol["LambdaStar0_j"][idx_s_test, idx_x_test, 0]

            if abs(base_val) > 1e-8:
                pct_change = (val - base_val) / base_val * 100.0
                pct_str = f"{pct_change:.1f}%"
            else:
                pct_str = "NA"
```

```python
            all_table_data.append(
                {
                    "State": f"(s={S[idx_s_test]:.1f}, x={X[idx_x_test]:.0f})",
                    "Scenario": name,
                    "Lambda_A": f"{val:.4f}",
                    "Abs Change": f"{val - base_val:.4f}",
                    "% Change": pct_str,
                }
            )

    df = pd.DataFrame(all_table_data)
    print("\n--- Pointwise Subsidy Effectiveness Comparison ---")
    print(df.to_string(index=False))


if __name__ == "__main__":
    run_part2()
```

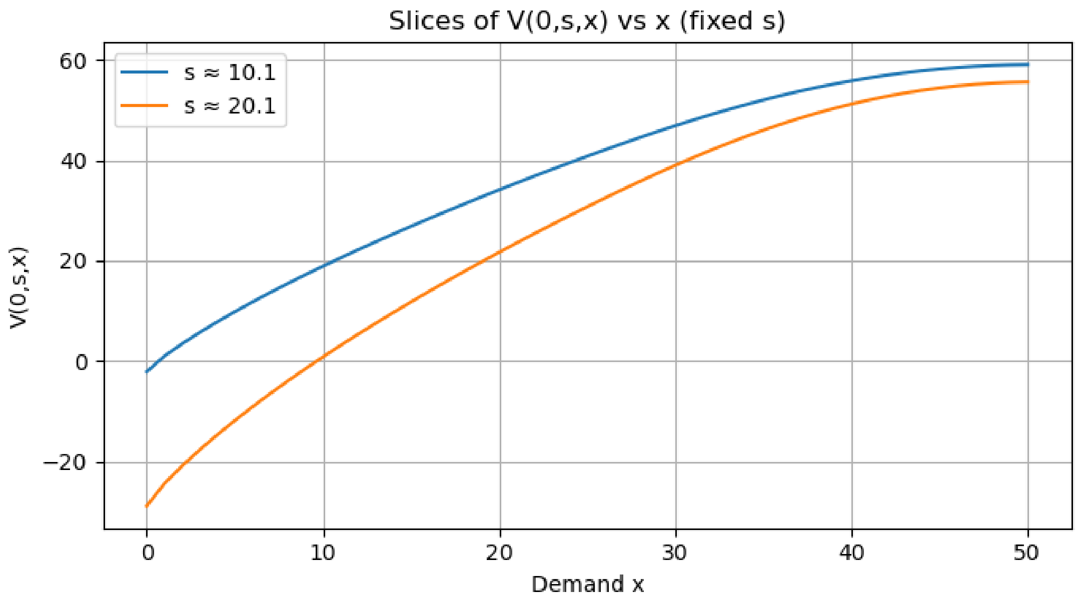
```
Solving HJB: T=5.0, r=0.02, _X=4.0
Grid: I=50, J=50, ds=1.0, dx=1.0
Integrating HJB backward in time…
Integration successful
```

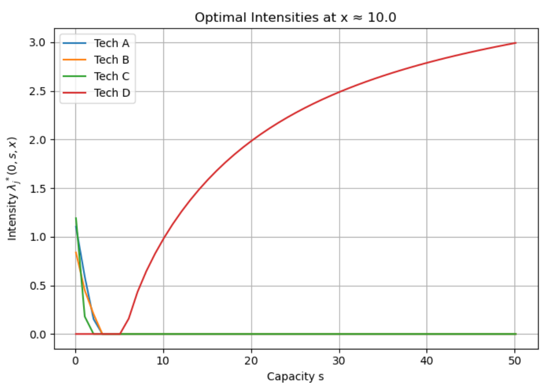


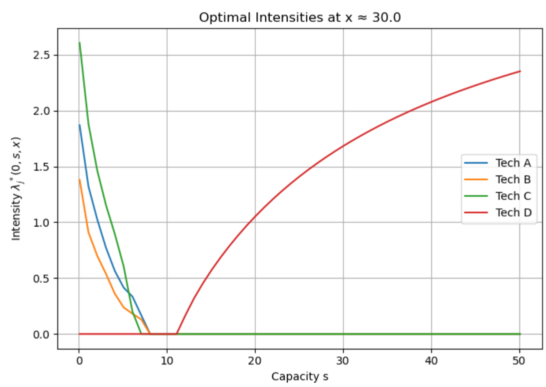
Value Function V(0, s, x)
Supply s
Demand x
V(0,s,x)



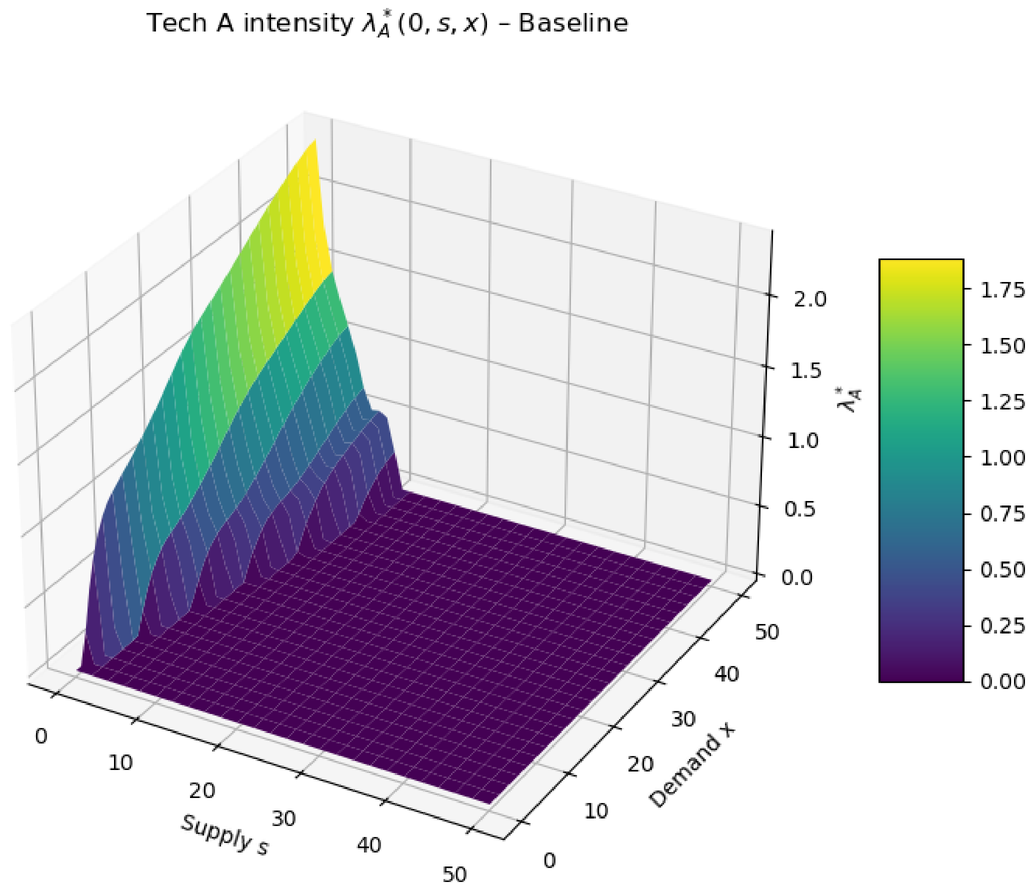
Slices of V(0,s,x) vs s (fixed x)
x ≈ 10.0
x ≈ 30.0
V(0,s,x)
Capacity s

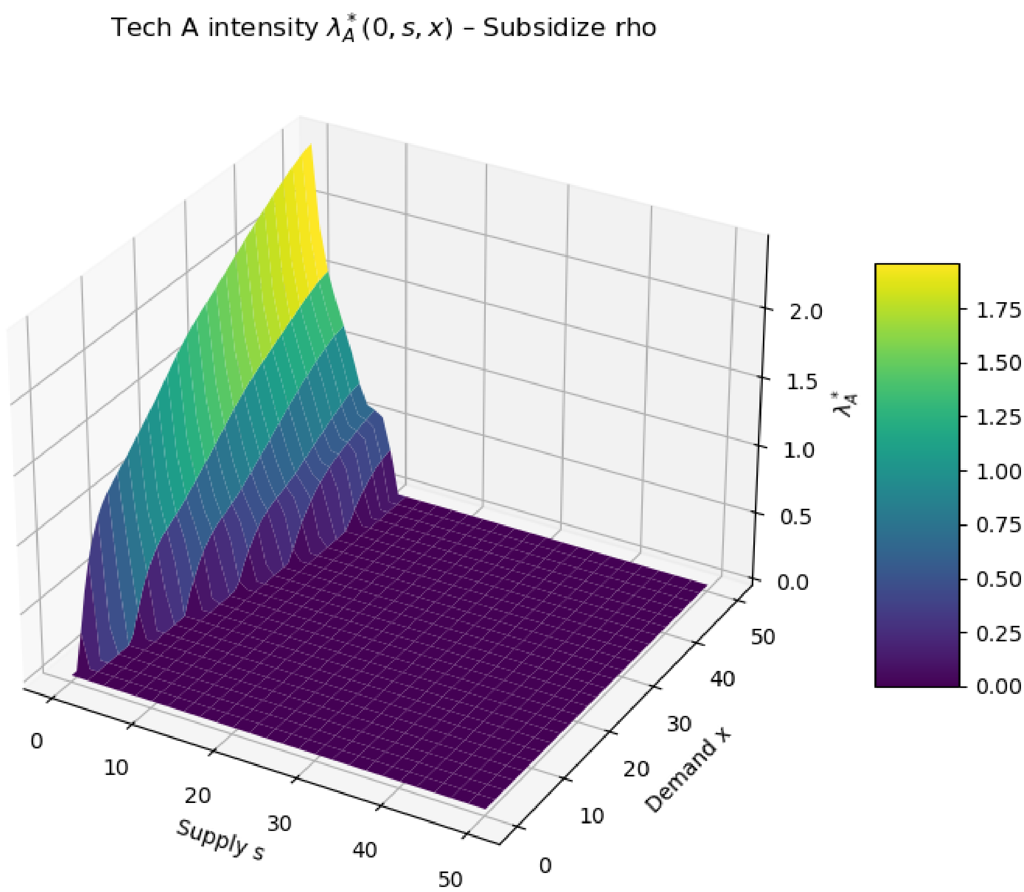


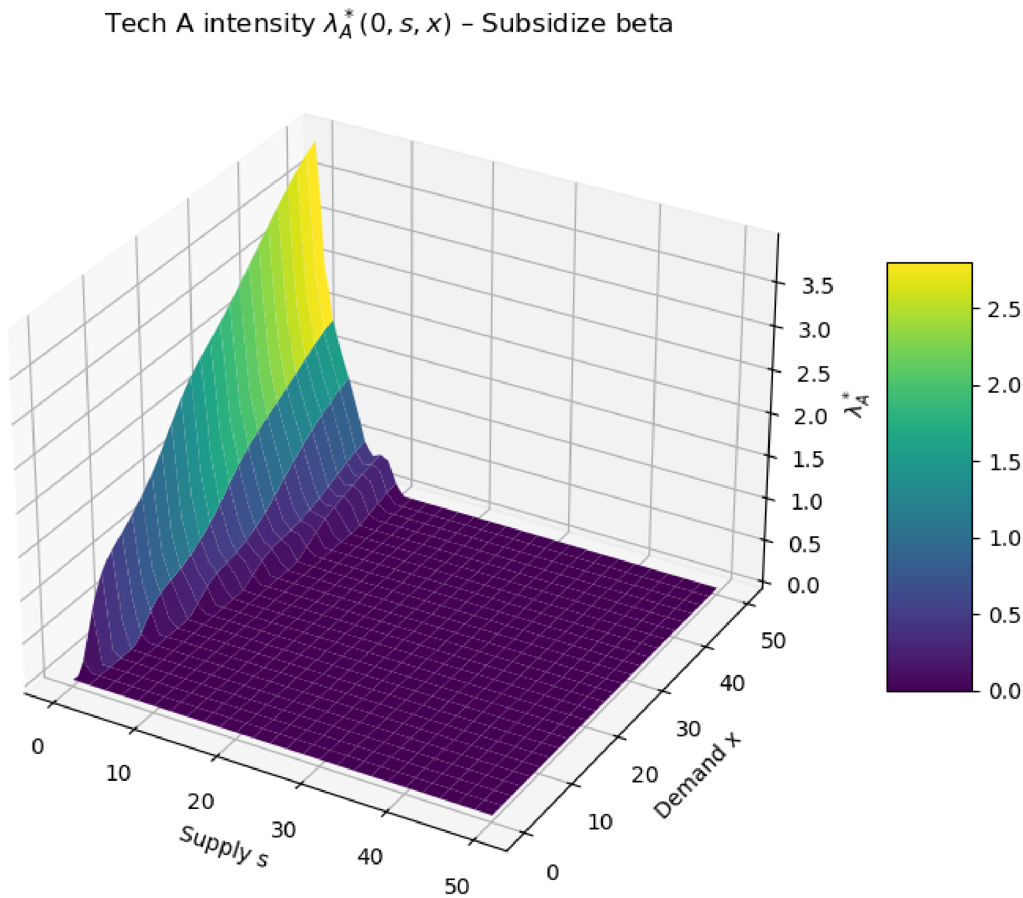


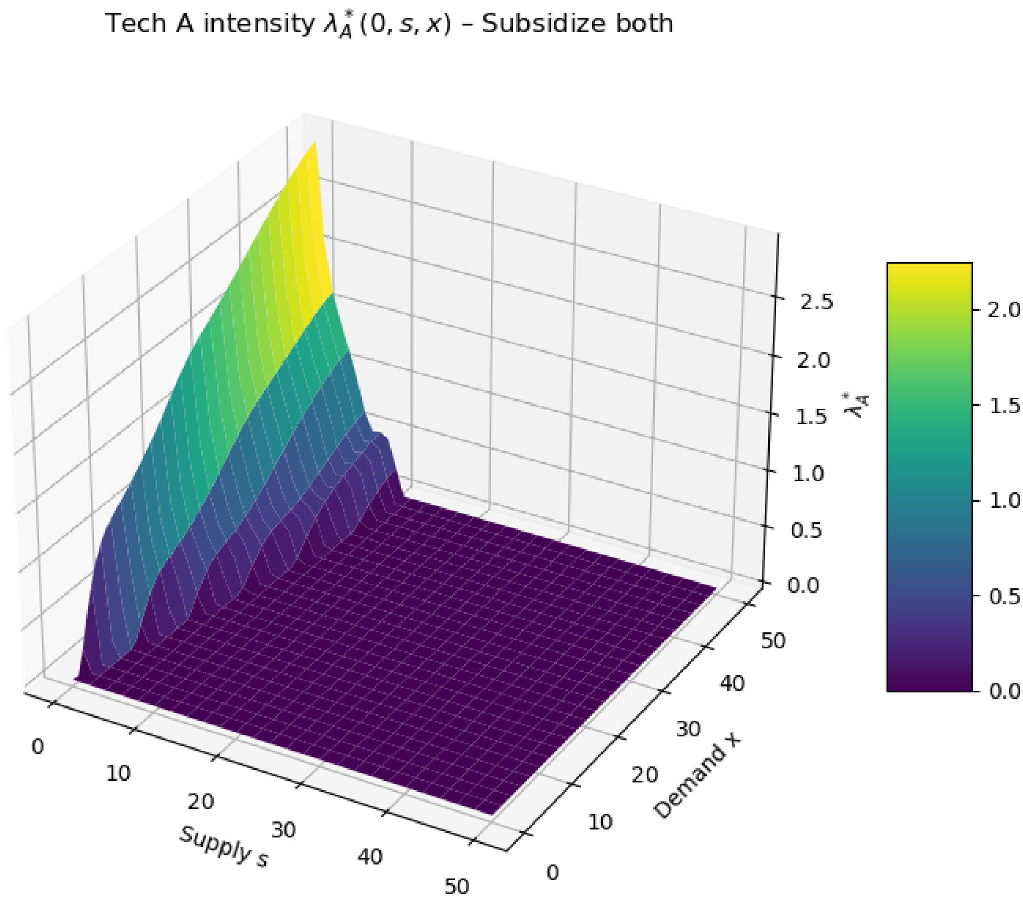


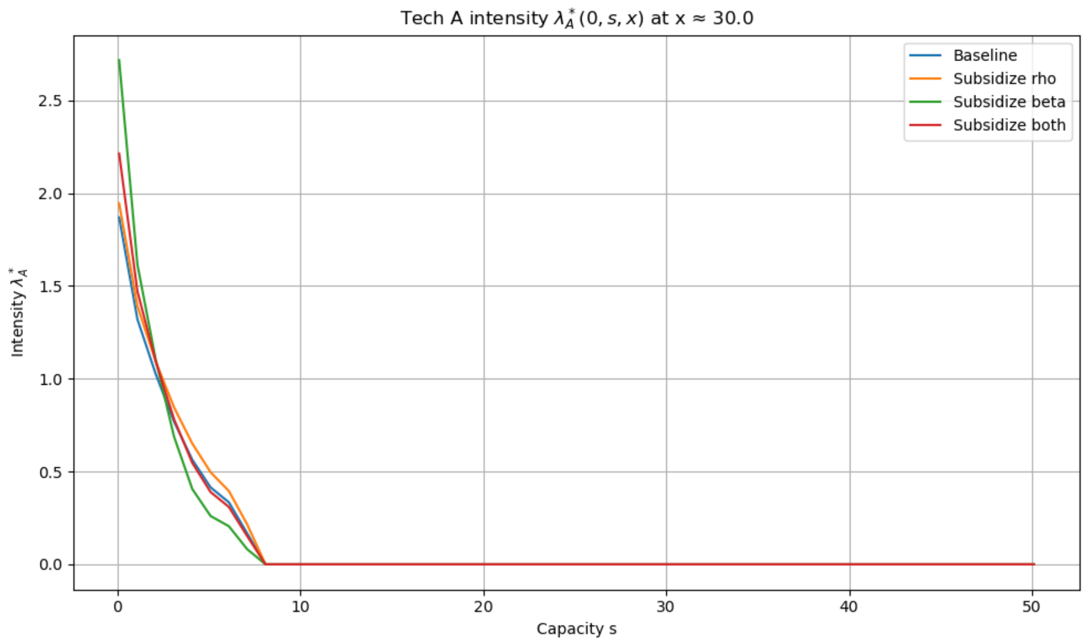

```
--- Policy Analysis ---
Solving scenario: Baseline
Solving scenario: Subsidize rho
Solving scenario: Subsidize beta
Solving scenario: Subsidize both
```


Tech A intensity $\lambda_A^*(0, s, x)$ – Baseline
Supply s
Demand x
λ_A^*
0
10
20
30
40
50
0.0
0.5
1.0
1.5
2.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75


Tech A intensity $\lambda_A^*(0, s, x)$ – Subsidize rho
λ_A^*
0.0
0.5
1.0
1.5
2.0
Supply s
0
10
20
30
40
50
Demand x
0
10
20
30
40
50
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75


Tech A intensity $\lambda_A^*(0, s, x)$ – Subsidize beta
Supply s
Demand x
λ_A^*
0
10
20
30
40
50
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5


Tech A intensity $\lambda_A^*(0, s, x)$ – Subsidize both
Supply s
Demand x
λ_A^*
0
10
20
30
40
50
0.0
0.5
1.0
1.5
2.0
2.5




```
Policy effectiveness at s=5.1, x=30.0
      Scenario Lambda_A Abs Change % Change
      Baseline   0.4145     0.0000     0.0%
 Subsidize rho   0.4962     0.0817    19.7%
Subsidize beta   0.2603    -0.1542   -37.2%
Subsidize both   0.3893    -0.0252    -6.1%

--- Pointwise Subsidy Effectiveness Comparison ---
        State       Scenario Lambda_A Abs Change % Change
(s=1.1, x=30)       Baseline   1.3205     0.0000     0.0%
(s=1.1, x=30)  Subsidize rho   1.3918     0.0713     5.4%
(s=1.1, x=30) Subsidize beta   1.6151     0.2947    22.3%
(s=1.1, x=30) Subsidize both   1.4705     0.1500    11.4%
(s=5.1, x=30)       Baseline   0.4145     0.0000     0.0%
(s=5.1, x=30)  Subsidize rho   0.4962     0.0817    19.7%
(s=5.1, x=30) Subsidize beta   0.2603    -0.1542   -37.2%
(s=5.1, x=30) Subsidize both   0.3893    -0.0252    -6.1%
(s=9.1, x=30)       Baseline   0.0000     0.0000       NA
(s=9.1, x=30)  Subsidize rho   0.0000     0.0000       NA
(s=9.1, x=30) Subsidize beta   0.0000     0.0000       NA
(s=9.1, x=30) Subsidize both   0.0000     0.0000       NA
```

0.0.2 Base Case Part 2

(a)

- The value function $V(0, s, x)$ is non-monotone with respect to capacity s. For a fixed level of demand x, the value initially increases as capacity grows from zero, reaches a distinct peak (the optimal capacity level), and subsequently decreases. At very high levels of capacity relative to demand (e.g., $s = 50$ when $x = 10$), the value actually becomes negative.
- Value of Additional Capacity: This indicates that the value of additional capacity is positive only when capacity is scarce. Once the capacity exceeds the optimal level for the current demand, the marginal value of additional capacity becomes negative. Adding supply beyond this point depresses the market price (P) so severely that the revenue loss on existing units outweighs the volume gain from the new unit.b)

(b)

- The value function $V(0, s, x)$ is strictly increasing with respect to demand x. Regardless of the current capacity level, an upward shift in demand always results in a higher firm value.

(c)

- We observe concavity in both dimensions: In s (Supply): The curve is concave (inverted U-shape), showing strong diminishing returns that eventually turn negative. In x (Demand): The curve is concave (increasing at a decreasing rate), reflecting the square-root relationship in the price function ($P \propto \sqrt{x}$).
- Economically, the curvature in capacity (s) illustrates market saturation. In this market, the firm is large enough that its capacity decisions affect the spot price. When s is low, capacity is scarce, prices are high, and the first few units of capacity generate immense value. As s increases, the spot price falls. Eventually, the revenue gained from selling one more unit is less than the revenue lost due to the price drop across all units. This creates a natural "limit" to profitable expansion, even if investment costs were zero. The negative values at high s imply that maintaining excessive capacity can be distinctively unprofitable (likely due to fixed operating costs or prices dropping below variable costs).

[ ]:

```

```

0.0.3 Base Case Part 3

(a)

- At $x \approx 10$, which technologies are deployed first? So, for that, at low levels of capacity ($s \approx 0$), Technology C is deployed with the highest intensity, followed by Technology A, and then Technology B.
- All technologies share identical cost structures ($\rho = 0.2, \beta = 2.0$), but they differ in their capital impact δ. Technology C has the highest impact parameter ($\delta_C = 5.0$), meaning it adds the most capacity per unit of investment intensity. Technology B has the lowest ($\delta_B = 2.0$).
- The firm behaves efficiently by prioritizing the technology that offers the highest marginal product of capacity for the marginal cost incurred. Therefore, the biggest mover (Tech C) is always utilized most aggressively when capacity is scarce.

(b)

- At $x \approx 30$, how does the ordering of technologies change, if at all? Are higher-δ technologies favored at higher demand levels? So, the rank ordering of technologies does not change. At $x \approx 30$, we still observe the hierarchy Tech C > Tech A > Tech B in the investment region. Expansion of Investment Region: While the ordering remains the same, the range over which these technologies are used expands significantly. At $x \approx 10$, investment stops around $s \approx 4$. At $x \approx 30$, investment continues until $s \approx 9$. Higher-δ technologies are consistently favored regardless of the demand level. Whether demand is low ($x = 10$) or high ($x = 30$), the firm always prefers the most efficient technology (Tech C) to close the gap between current capacity and target capacity. High demand does not change the relative preference between technologies, it simply increases the total volume of investment required, scaling up the intensities of all productive technologies while maintaining their relative efficiency ranking.

[ ]:

0.0.4 Policy Implication Part 2

ρ - Lowering the fixed/linear cost proves to be consistently effective across different investment regimes. At the aggressive early stage ($s = 1.1$), it increases intensity by 5.4%, and at the moderate stage ($s = 5.1$), it increases it by 19.7%. This subsidy works uniformly by raising the net marginal benefit of every unit of investment, regardless of the current scale.

β - This subsidy displays a regime-dependent behavior. In the Rapid Growth phase ($s = 1.1$): Where baseline intensity is high ($\lambda > 1$), reducing convexity is the most powerful policy, boosting intensity by 22.3%. It effectively rewards mega-projects by reducing the penalty for scaling up intensity. In the Moderate phase ($s = 5.1$): The intensity drops by 37.2%. Because $\lambda < 1$ in this region, the change in the exponent dominates, shifting effort away from this maintenance phase back toward the start of the horizon.

both - This scenario ($\rho = 0.15, \beta = 1.8$) illustrates the interaction between the two levers. It functions as a middle ground policy. At $s = 1.1$, it successfully boosts investment (+11.4%), though not as aggressively as the pure convexity subsidy. At $s = 5.1$, it significantly mitigates the downsides of the β subsidy. While the pure β-subsidy caused a steep 37.2% drop in intensity, the combined approach limits this drop to just 6.1%.

So, a balanced policy portfolio helps avoid the boom-bust nature of purely targeting convexity. It captures some of the early-stage acceleration benefits while using the linear subsidy (ρ) to maintain a floor for investment incentives during the moderate expansion phase. (All lines have flattened out to 0 by $s = 9$. This confirms that once capacity is sufficient ($s > 8$), investment stops regardless of the subsidy.)

[ ]:

0.0.5 Policy Implication Part 3

- Reducing ρ (Linear Subsidy): This scenario closely resembles Investment Tax Credits or Feed-in Tariffs. These policies essentially pay a fixed amount per unit of capacity or energy, directly lowering the effective linear cost (ρ) or increasing the linear revenue. They are effective at pushing projects over the profitability hurdle across a wide range of conditions.

- Reducing β (Convexity Subsidy): This scenario behaves like policies that reduce soft costs (permitting, interconnection, supply chains). Our results at $s = 1.1$ demonstrate that these policies are uniquely suited for jump-starting an industry. They allow developers to build massive capacity quickly (the green spike in the plot). However, they are less effective at sustaining smaller, marginal investments later in the lifecycle ($s = 5.1$).
- So, most real-world policies (like the US Inflation Reduction Act's tax credits) behave more like reducing ρ. They provide linear financial incentives that lower the base cost of entry. However, as the renewable transition demands faster build-outs (scaling from GW to TW), policymakers are increasingly looking at β-reducing policies to allow for the massive surges in capacity seen in the green line of our plots. So, while ρ-subsidies (like the Inflation Reduction Act's tax credits) provide a broad baseline of support, policymakers aiming for extremely rapid decarbonization (GW to TW scale) must also target β reductions to unlock the high-intensity build rates seen in the early states of our model.

[ ]: